MODULE-6 JENKINS AND BUILDING TOOLS:

**✅ VS Code me SSH terminal kaise open karein (GREEN prompt ke saath) ya command line interface:**

**Ye command chalao 👇total 3 step hai**

Step1:

dir $HOME

Aur phir:

dir $HOME\Downloads

yeh command chalany k baad jo bhi humnay key pair bnaya current ec2-instance ka agar mil jata hai

then.

Step2:VS Code me ye dono command ek baar chalyoh:

1.)keypairnameofcurreninstance

command:

icacls C:\Users\sagar\Downloads\keypairname /inheritance:r

icacls C:\Users\sagar\Downloads\keypairname /grant:r sagar:R

👉 **Is command ka matlab:**  
ec2.pem file par sirf user sagar ko read permission di gayi hai aur baaki sab users ki inherited permissions hata di gayi hain, taaki file secure rahe. 🔐

humay ec2 insatnce public ip m dikkat aari thi bcz ec2 m yeh real public ip nahi hoti due to which browser/ssh open nahi hota so we use elastic ip.

elastic ip is permanent public ip srever k restart/stop honay per bhi change nhi hoti Jenkins production and vs m ssh k liye must hai

Elastic IP: 54.157.68.118 this is our elastic ip for ex

Ab hum aws m k ec2 m search karengay elastic ip then usay current instance k saath allocate karenagay elastic ip search karkay allocate karoh then associate karenagy current instance jo running hai select karengay

**✅ Elastic IP associate with EC2**

AWS Console → Elastic IP → Associate with instance

**Result:**

Public IPv4 address: 54.157.68.118

👉 Ab ye **real public IP** hai.

👉 Ye step vscode Windows poweshell me SSH ke liye important hota hai.tabhi ssh connect hogaa

✅ Step 3: Ab SSH connect karo (FINAL COMMAND)

ssh -i C:\Users\sagar\Downloads\ec2.pem ubuntu@elastic-ip of current instance

Example:

ssh -i C:\Users\sagar\Downloads\ec2.pem ubuntu@54.157.68.118

then our ssh open << agar ssh stuck hojata hai ya apnay connection tut jaye toh ec2 instance ko phele stop kardoh 2 min baad start kardoh ab status m dekho 10 min ruko jab green color may 3/3 check pass ajay phir yeh tino step chalyoh tab ssh hojeyga >>

Aur agar command line interface ssh say exit karna hai toh 2 baar exit type kardoh like:

Exit , exit

Agar hum pta karna chathay hai public ip toh use :

\*curl ifconfig.me\*

agar dubara connection time out aye toh yeh hua

**❌ Kya problem thi?**

**1️⃣ EC2 “Instance Reachability Check” fail ho gaya tha**

Iska matlab:

* EC2 ka **operating system hang / freeze** ho gaya tha
* Network services (SSH, ping) **response nahi de rahi thi**

Isliye tumhe:

* ❌ SSH → connection timed out
* ❌ Ping → request timed out

Mil raha tha.

**2️⃣ Iska key / Elastic IP / command se koi lena-dena nahi tha**

* .pem file ✔️
* Elastic IP ✔️
* SSH command ✔️

Problem **instance ke andar** thi.

**✅ Kaise thik hua?**

**🔄 Tumne Stop → Start kiya**

Isse kya hua:

* EC2 ka OS **fresh boot** hua
* Network stack reset ho gaya
* SSH service dobara properly start ho gayi

Result:

* ✅ **3/3 status checks passed**
* Instance healthy ho gaya
* Vaapis pingTop of Form

Bottom of Form

54.157.68.118 kiya toh eror ara tha but kio baat nhu phir bhi ssh vaali commands chalyoh ssh chal jeyega

ssh -i C:\Users\sagar\Downloads\ec2.pem [ubuntu@54.157.68.118](mailto:ubuntu@54.157.68.118) ssh opsen

🔹 Jenkins ky hai & kyun use hota hai?

Jenkins ek automation tool hai jo software ke kaam ko automatically karta hai jaise:

* Code build
* Test run
* Deploy and

Jenkins developer ke code ko bar-bar manually check karne ki jagah automatic pipeline bana deta hai.

👉 Kyun use hota hai?

* Manual kaam kam hota hai
* Fast feedback milta hai , error jaldi pakad me ajat hai , save time
* 📦 Software Company Example (Industry)

Company: E-commerce Website

❌ Without Jenkins

* Developer manually deploy karta
* Night deployment
* Website down risk 😰

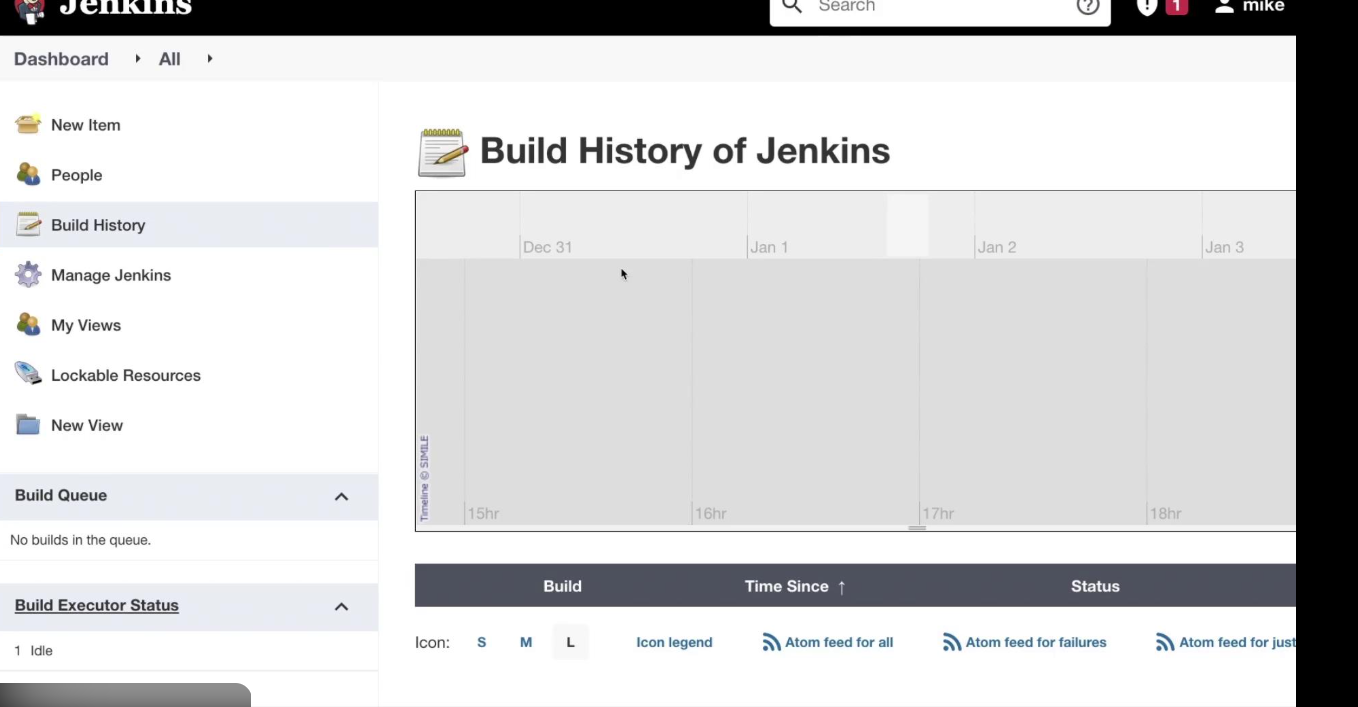
✅ With Jenkins

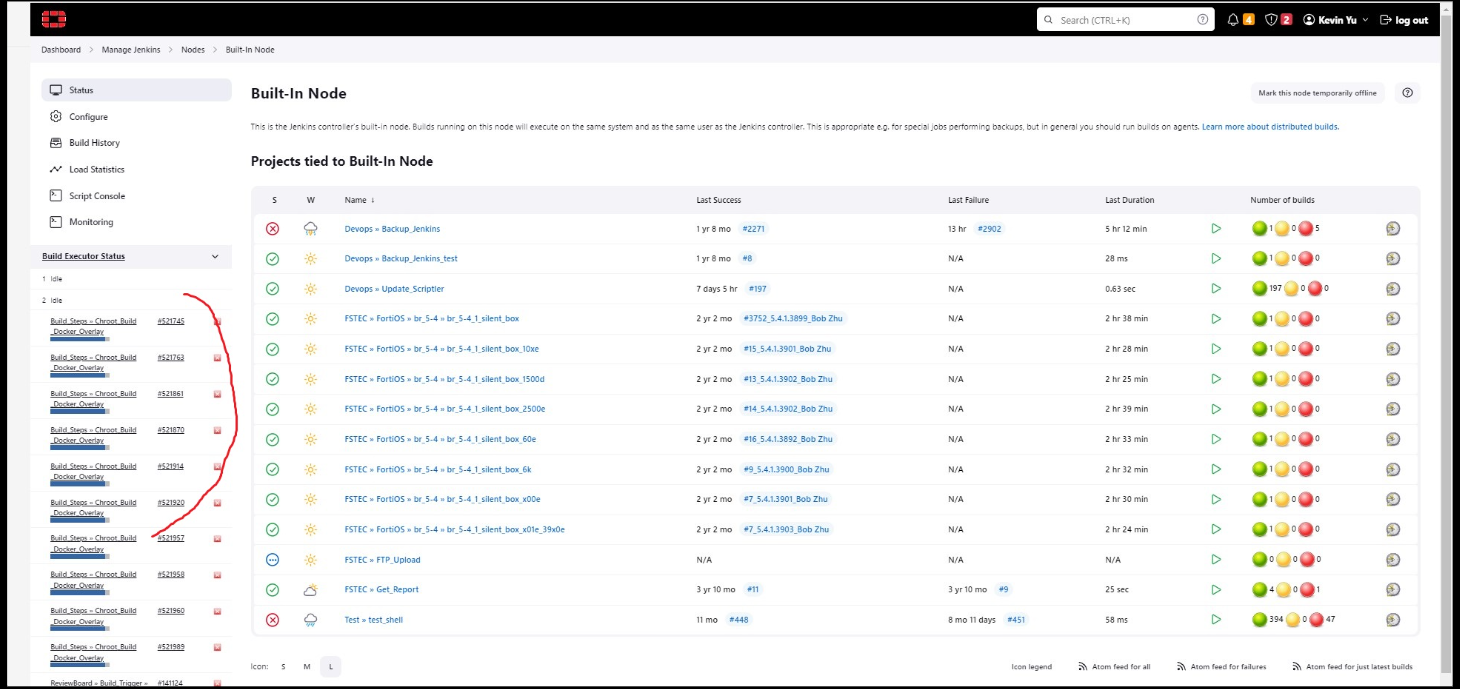
1. Developer → git push
2. Jenkins automatically:
   * Build
   * Test
   * Deploy
3. Website update within minutes 🚀

Real life ex 🔔 Alarm Clock = Jenkins

| Alarm Clock kya karta hai: |
| --- |
| Time pe uthata |
| Automatic ring |
| Tumhe yaad nahi rakhna |

|  |
| --- |
|  |
| So Jenkins bhi clock jaisay kaam karta hai:  Like: code change detect , automatic build , developer want to deploy then Jenkins deploy kardega automatically. |
| >>What is CI(continuous integeration)?  Ans: integrated code means: Alag-alag developers ka code ek saath jod diya jana  “Continuous Integration(CI) is a practice where developers frequently integrate code into a shared repository and each integration is automatically built and tested.”  matlab ki: jaise hi developer code push karta hai, system automatically build aur test karta hai.  Why we use ci?  >early bug detection  >less conflict  >automation  >safer production , team code stable  CI KA EXAMPLE  🧠 Continuous Integration (CI) – Easy Example  🔹 Project: Login Website  Team me 2 developers hain: Dev-1 aur Dev-2 CI tool: Jenkins  Dev-1 ka kaam (Login feature);  1️⃣ Dev-1 login ka code likhta hai 2️⃣ Code GitHub me push karta hai:  git add .  git commit -m "Login feature added"  git push origin main  3️⃣ Jaise hi code push hota hai:   * Jenkins automatically start ho jata hai * Code pull karta hai * Build karta hai * Tests run karta hai   ✅ Result:   * TEST PASSED 👉 Login code safe hai   ➡️ Ye process = Continuous Integration  Dev-2 ka kaam (Signup feature):  1️⃣ Dev-2 signup ka code add karta hai 2️⃣ Push karta hai:  git add .  git commit -m "Signup feature added"  git push origin main  3️⃣ Jenkins phir se: code pullkarega ,buildkarega ,testrunkarega  ❌ Result   * TEST FAILED   👉 Matlab:   * Signup code se login break ho gaya * Dev-2 ko turant pata chal gaya * End me bada issue nahi bana   🔄 Is poore flow ko hi CI kehte hain  Code baar-baar push hota hai aur system har baar automatically build & test karta hai  ❓ Agar Dev-2 ko turant pata nahi lagta, to kab pata lagta?  🔴 Bina CI ke (Worst case)  Dev-2 ko late pata lagta, jaise:  1️⃣ Deployment ke time   * Jab website server par upload hoti * Ya production me jaati   👉 Tab error dikhta ❌ Customer already affected  🧠 CI me mainly 3 jagah commands use hoti hain:  1️⃣ Developer machine (VS Code terminal) 2️⃣ CI server (Jenkins / EC2 terminal) 3️⃣ Application test/build commands  1️⃣ Developer side commands (VS Code Terminal me)  👉 Ye commands developer khud chalata hai 👉 Purpose: code ko repository me bhejna  🔹 Commands:  1.) git add .  2.) git commit -m "login feature added"  3.) \*git push origin main\* : local code ko githubper push karta hai and Local code means jo code developer ke apne laptop  🔹 Output (example):  [main 3f2a9c1] login feature added  2 files changed, 25 insertions(+)  >> CI(continuous integeration) flow:    >>WHAT IS CD(CONTINOUS Deployment /CONTINOUS DELIVERY ?  ANS: where code changes are automatically deployed on a server after successful build and testing.  🔹 Matlab: CI ke baad code automatically server par chala jata hai.   **Continuous Delivery:** after successful code testing उसे लाइव करने के लिए एक 'Manual Trigger' (इंसानी मंजूरी) की जरूरत होती है।   **Continuous Deployment:** इसमें किसी भी मानवीय हस्तक्षेप की आवश्यकता नहीं होती। यदि कोड सभी टेस्ट पास कर लेता है, तो वह अपने आप लाइव हो जाता है।  ❓ CD kyun use kiya jata hai? (Why we use CD)  1️⃣ Manual deployment ka jhanjhat khatam 2️⃣ Fast delivery – feature jaldi live hota hai 3️⃣ Human error kam – sab automatic 4️⃣ Reliable deployment – same process har baar 5️⃣ Customer ko fast update milta hai  👉 CI check karta hai code sahi hai ya nahi 👉 CD ka kaam hai code ko server par bhejna  🧠 CI + CD ka relation (short me)   * CI = Build + Test * CD = Deploy (Server par chala dena)   🌍 CD ka Real-world Example  🔹 Project: Login Website  CI tool: Jenkins Server: AWS EC2  📌 Step-by-Step Flow:  1️⃣ Developer code push karta hai  Use \*git push origin main\* then  2️⃣ CI start hota hai   * Code pull * Build * Test   ✅ Tests PASS ho gya  3️⃣ CD automatically start hota hai   * Code server (EC2) par copy hota hai * Application restart hoti hai   4️⃣ Website LIVE ho jaati hai 🌐  👉 Developer ko manually kuch nahi karna  ❌ Bina CD ke kya hota?   * Developer ko:   + Server login karna   + Files copy karni   + App restart karni   ❌ Time waste ❌ Galti ka chance zyada  >>Commands for cd(continuous deployment):  **✅ STEP 1: Pehle ek change banao (VS Code editor me)**  1️⃣ VS Code me **koi file open** karo (example: README.md ya koi .txt / .html file)  2️⃣ Last me ek line add karo:  CI/CD practice line  3️⃣ **Save** karo (Ctrl + S)  **✅ STEP 2: Ab VS Code terminal me push karo**  git status  👉 Output aisa hoga:  modified: README.md  Ab commands:  \* git add . \* avoide star  \* git commit -m "test change for pull" \*  \* git push origin main \*  Expected output:  [main a1b2c3d] test change for pull  1 file changed, 1 insertion(+)  **⚠️ IMPORTANT BAAT (yaad rakhna)**  👉 **Isi folder me git pull chalane par fir bhi:**  Already up to date.  aayega ❌ Ye error nahi hai ❌ Ye normal hai  important command: ls -Force >yeh hidden file dikha deta hai and .git m honi chayie  **📊 CI vs CD – Main Differences Table:**   | **Point** | **CI (Continuous Integration)** | **CD (Continuous Deployment)** | | --- | --- | --- | | Main kaam | Build + Test | Deploy | | Start kab hota | Code push par | CI success ke baad | | Purpose | Bug jaldi pakadna | Code ko live banana | | Automation | Testing tak | Deployment tak | | Risk | Low | Thoda high (live impact) | | Tools | Jenkins, GitHub | Jenkins, ArgoCD, AWS | | Output | PASS / FAIL | Application LIVE |     **🔹 CI aur CD ka RELATION (Most Important ⭐⭐⭐):**  1️⃣ **CI CD ka base hai** 👉 CI fail → CD start hi nahi hota  2️⃣ **CI quality check karta hai** 👉 CD usi quality code ko deploy karta hai  3️⃣ **CI without CD possible hai** 👉 Lekin **CD without CI risky** hota hai ❌  4️⃣ **CI + CD = Complete automation pipeline**  **🔁 Flow samjho (One line)**  **Code push → CI (build + test) → CD (deploy) → App LIVE**  >>Jenkins industry m kaisy and kha use hotahai:  **1️⃣ Continuous Integration (CI) – Sabse common use case**  **Industry me kaise use hota hai:**  Bohot saari jagah use hota hai inmay say kuch points .   * Developer code push karta hai * Jenkins automatically:   + Code pull karta hai   + Build karta hai   + Tests run karta hai   👉 **Benefit:** Bugs jaldi mil jaate hain  **2️⃣ Continuous Deployment (CD):**  **Industry use:**   * CI pass hone ke baad * Jenkins automatically:   + Code server (EC2) par deploy karta hai   + Application restart karta hai   👉 **Benefit:** Manual deployment ki zarurat nahi  **3️⃣ Automated Testing**  **Companies me use:**   * Unit tests * Integration tests * Regression tests   👉 Jenkins har build ke saath test chalata hai 👉 **Benefit:** Quality high rehti hai  **4️⃣ Nightly / Scheduled Builds**  **Industry scenario:**   * Daily raat ko build & test * Jenkins scheduler use hota hai   👉 **Benefit:** Subah team ko report mil jaati hai  **5️⃣ DevOps Automation**  **Jenkins se automate karte hain:**   * Docker image build * Kubernetes deployment * Infrastructure scripts run   👉 **Benefit:** Fast + reliable DevOps flow  **6️⃣ Code Quality Check**  **Industry me kaise: Jenkins&sonarqube say jo code kay**   * Changes ko, bugs, and security issues check kartay hai   👉 **Benefit:** Production-ready code |
|  |





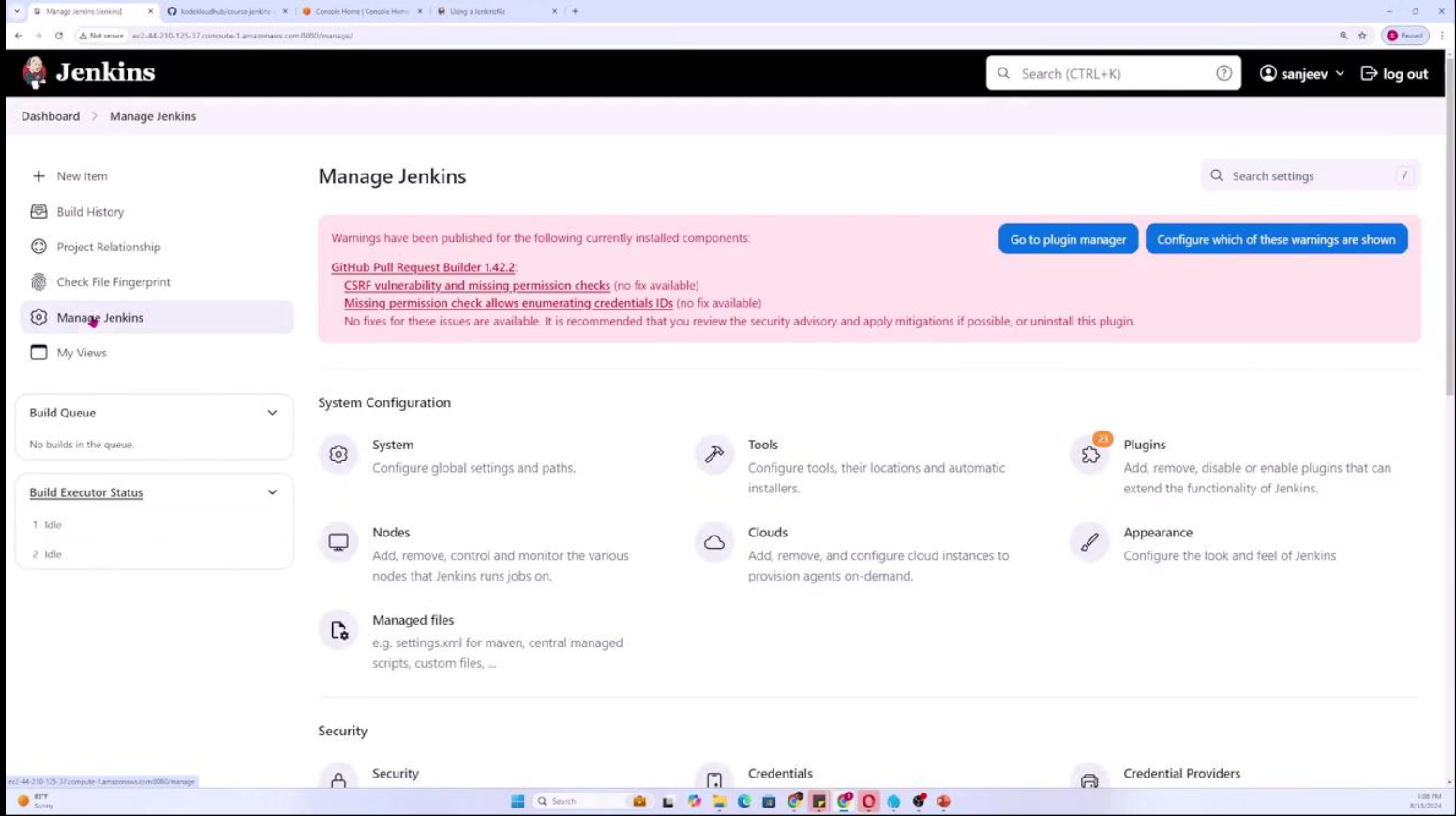
**🧭 Jenkins Dashboard ko image se kaise pehchaano (mapping):**

Image dekhte hue ye match karo 👇

**Dashboard** → Jobs ko ek jagah dekhne, monitor karne aur run karne ka main screen

* **New Item** → Left side menu me (naya job/pipeline banane ka option)
* **Build History** → Left panel me recent builds (green/red)
* **Job List (Project List)** → Center table me saare jobs ke naam
* **Build Status (Green/Red)** → Job ke saamne color icon
* **Build Queue** → Left side “Build Queue” section (waiting jobs)
* **Build Executor Status** → Left side “Build Executor Status” (Idle/Busy)
* **My Views** → Left menu me jobs ko category/view me dekhne ke liye
* **Manage Jenkins** → Left menu ka main admin control
* **Credentials (via jobs)** → Job config ke andar credentials selection
* **User/Profile** → Top-right corner (username + logout)

Yeh hmara manage Jenkins hai:



**🧭 Manage Jenkins – One-line Explanation**

* **System** → Jenkins ke global settings (URLs, env vars, email, paths) set karne ke liye
* **Tools** → Git, Maven, Java, Node jaise tools ki location/config set karne ke liye
* **Plugins** → Jenkins ki functionality add/remove/enable/disable karne ke liye
* **Nodes (Agents)** → Jenkins agents add/manage karke jobs distribute karne ke liye
* **Clouds** → Cloud se on-demand agents provision(phele say kuch arrange karkhay rakhna) karne ke liye
* **Appearance** → Jenkins UI ka look & feel customize karne ke liye
* **Security** → Authentication, authorization, permissions set karne ke liye
* **Credentials** → Git, server, cloud ke secrets (keys/passwords) securely store karne ke liye
* **Credential Providers** → Credentials ke sources/types configure karne ke liye
* **Users** → Jenkins users create/manage karne ke liye
* **System Log** → Jenkins logs dekhne/troubleshoot karne ke liye
* **Reload Configuration** → Jenkins restart kiye bina config reload karne ke liye
* **Manage Plugins (Advanced)** → Plugin updates/versions/advanced options ke liye
* **About Jenkins** → Jenkins version & environment info dekhne ke liye

Day-2 jenkins

3feb 2026

Top of FormBottom of Form>>Today we learn how to install Jenkins:

Jenkins install karnay k liye hum ec2 m security group m 8080 port custom tcp m inbound rule save karengay then hum ek new instance bnayengay t3 micro m

Hum ec2 terminal nahi karengay hum vs code m Jenkins install kaengay hmay bohot dikkat aari thit oh hum Jenkins ko snap say download karengay. And hum elastic ip ka use karengay elastic ip permanent hoti hai ec2 vaali ip temporary hoti hai.

First we start ssh in vs code so we use these comands

(1)dir $HOME\Downloads

icacls C:\Users\sagar\Downloads\

(keypairname-of-currentinstance) /inheritance:r

icacls C:\Users\sagar\Downloads\keypairname /grant:r sagar:R

ex:

command:

icacls C:\Users\sagar\Downloads\keypairname /inheritance:r

icacls C:\Users\sagar\Downloads\keypairname /grant:r sagar:R

\=path and /=option

(4) ssh -i C:\Users\sagar\Downloads\ec2.pem ubuntu@54.157.68.118

then our ssh open 54.157.68.118 yeh elastic ip hai

agar kabhi jab hum sab close karkay dibara connect kartay hai ssh toh agar iska status 2/3 ajata toh iskay instance ko stop kardoh wait karoh jab tak stop likha nahi ajat the vaapis start karoh uskay baad status 3\3 aajeyag then command chalyoh vs code m aur yeh elastic ip kuch bhi ho skati but current instance ki yeh hai

ssh -i C:\Users\sagar\Downloads\ec2.pem ubuntu@54.157.68.118

then ssh open ho jayega

**✅ PRACTICAL & INDUSTRY SOLUTION (Recommended) jenkins installation**

**⭐ OPTION 1 — SNAP se Jenkins install karo (BEST & CLEAN)**

💡 Industry me jab repo signing issue ho, **snap / docker** use kiya jata hai.

**🚀 SNAP METHOD (Guaranteed working)**

**STEP 1: Snap install (agar nahi hai)**

sudo apt update

sudo apt install snapd -y

**STEP 2: Jenkins install via snap**

sudo snap install jenkins –classic

step3: Jenkins snap status check:

sudo snap services Jenkins.

**Expected output:**

jenkins 2.xxx installed

step3: Jenkins snap status check:

sudo snap services Jenkins.

**>>Jenkins stop karnay k liye use kar saktay hai:**

sudo systemctl stop snap.jenkins.jenkins

**>>now type this command:**

sudo nano /var/snap/jenkins/common/config.xml

then editor khulega usmay

(agar kuch aur hai toh replace kar do): uskay baad yeh sara paste kardoh avoid bracktes.

[

<?xml version='1.1' encoding='UTF-8'?>

<hudson>

<useSecurity>false</useSecurity>

</hudson>

] avoid these beracket[]

👉 **Bas itna hi:**  
👉 Username / password ka koi role nahi

CTRL + O → Enter

CTRL + X

**🔥 then Jenkins start:**

sudo systemctl start snap.jenkins.jenkins

**Ab browser me open karo:**

http://<EC2-PUBLIC-IP>:8080

**humne Java manually install nahi ki**lekin Jenkins (Snap) ne Java khud install kar li but updated version nahi karta humay dalna hoga updated which is jdk17.

First we refresh karengay snap Jenkins ko using:

>> sudo snap refresh Jenkins

Then Jenkins restsrt use

>> sudo snap restart Jenkins

then Jenkins ka java version check karenagy:

>> sudo snap run jenkins.java -version

expexted version17 jenkins k liye java 17 best hai

now method 2: ismay hum Jenkins ko force karengay custom java version k liye isay tab use kartay hai jab humay specific version chaiy hota hai

>> sudo apt update

>> sudo apt install openjdk-17-jdk -y

for check >> java -version

then hum Jenkins ko java use karnay k liye bolengay using command;

>.sudo snap set jenkins java.home=/usr/lib/jvm/java-17-openjdk-amd64

then use:

>>sudo snap restart Jenkins and use >>sudo snap run Jenkins.java-version

>>for checking status of Jenkins:

>> sudo systemctl status snap.jenkins.jenkins

agar command not found ho jaye toh yeh chalyoh

>> sudo snap services Jenkins.

Confirm Jenkins snap install hai ya nahi:

>> snap list | grep jenkins

🔹 Expected output

jenkins 2.xxx latest/stable jenkinsci✓ then

>>: Jenkins auto-start enable (MOST IMPORTANT):

sudo systemctl enable snap.jenkins.jenkins

**🔹 Expected output**

Active: active (running)

agar output active running aye toh that ok if error then manually start/restart command chalyoh.

Now hum sikhengay manually kaisy start kray Jenkins ko:

**✅ step 1: Jenkins manually start / restart:**

sudo systemctl start snap.jenkins.jenkins

sudo systemctl restart snap.jenkins.jenkins

**👉 Jenkins manually start / restart bhi kar sakte ho.**

**Then Vaapis check status using:**

>> sudo systemctl status snap.jenkins.jenkins

**>> Actual snap revision folder dekhna hai toh use yeh chaiy hota hai number humay:**

Har snap ka **revision number alag hota hai** (4983 fix nahi hota). Then we verify using this commands:

ls /var/snap/jenkins/

**Output example:**

common 5120 current

👉 Yahan jo **number** dikhe (jaise 5120) wahi real folder hai.

Check Jenkins 8080 port par listen kar raha hai ya nahi:

sudo ss -tulnp | grep 8080

🔹 Expected output

LISTEN 0 50 \*:8080 users:(("java",pid=XXXX))Top of Form

That means Jenkins backend ready

**now: Jenkins browser me open karna**

**👉 Kab use karein?**

* Jenkins UI access karne ke liye

**🔹 URL**

http://<ELASTIC-IP>:8080

Example:

<http://54.157.68.118:8080>

Agar error aye toh that means

Browser me http://3.93.185.211:8080 **timeout** aa raha hai, iska matlab **Jenkins web service ya to chal hi nahi rahi, ya port 8080 bahar se open nahi hai**.

Main seedha **reason → check → fix** bata raha hoon.

**❌ Ye error kyun aa raha hai?**

ERR\_CONNECTION\_TIMED\_OUT

Iska matlab **browser EC2 tak pahunch gaya**, lekin:

1. ❌ Jenkins service running nahi
2. ❌ Jenkins 8080 par listen nahi kar raha
3. ❌ EC2 Security Group me 8080 allow nahi
4. ❌ Local firewall (UFW) block kar raha

👉 Inme se **koi ek bhi** ho sakta hai.

So ek nya instance bna loh safe zone.

Agar open ho jata hem

Bottom of Form

**agar open hogya then for: Initial password:**

sudo cat /var/snap/jenkins/4983/secrets/initialAdminPassword

**Example output:**

a7f3c9d4e2b64c9f8a1b0c3d4e5f6a7b

Is output ko copy karkay password m daal doh

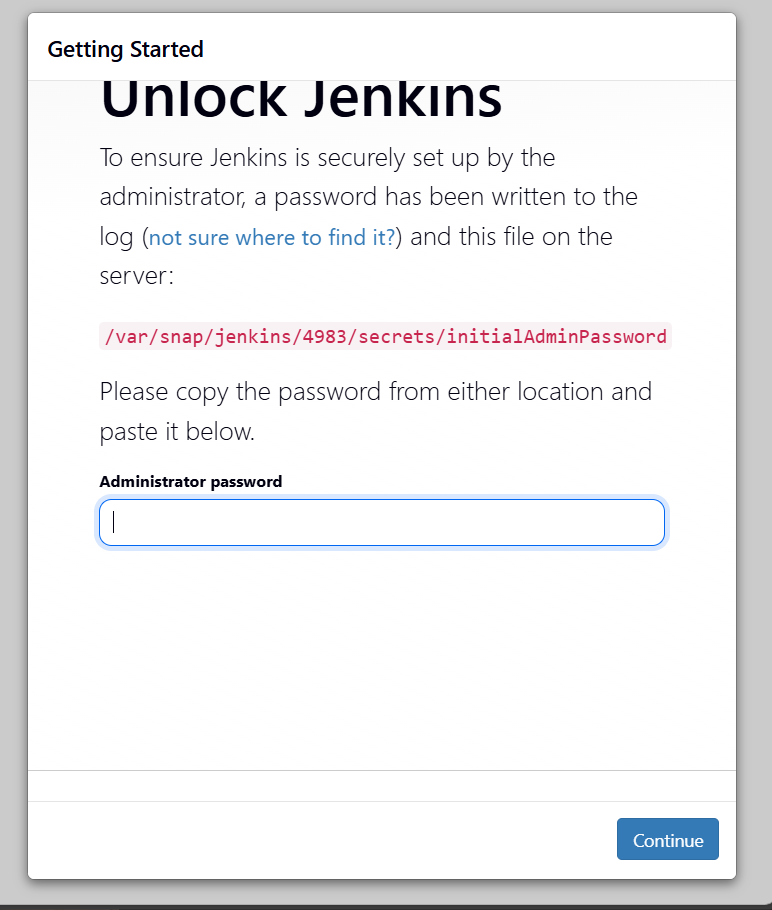
**Jenkins page me paste karo**

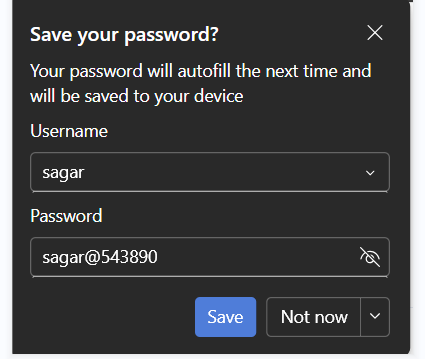
* Jenkins browser page
* **Administrator password** box
* Paste → **Continue**

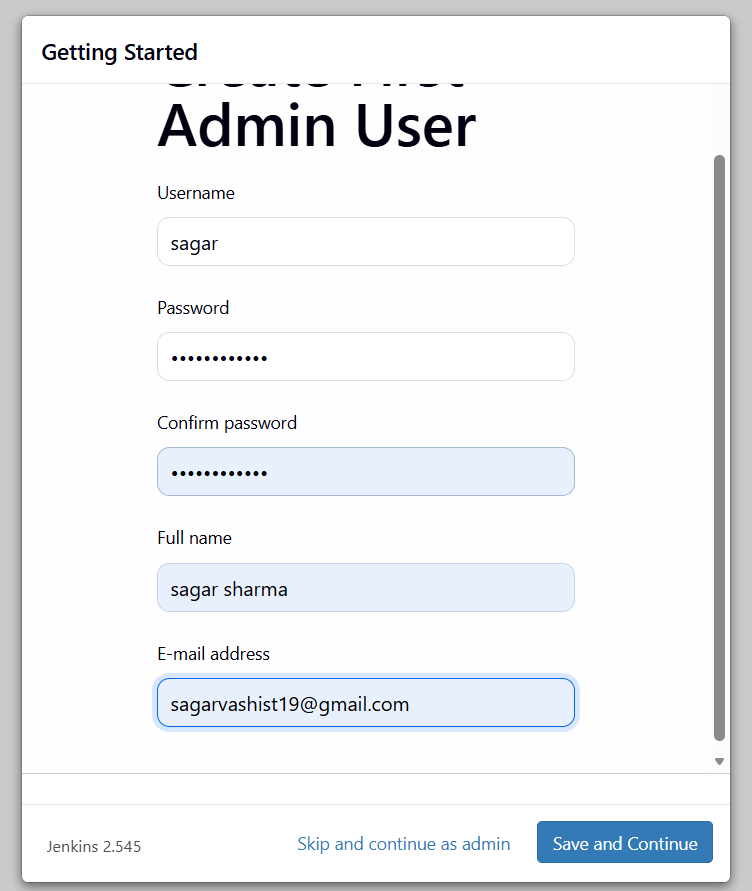
**🔍 Extra check (ensure same instance)**

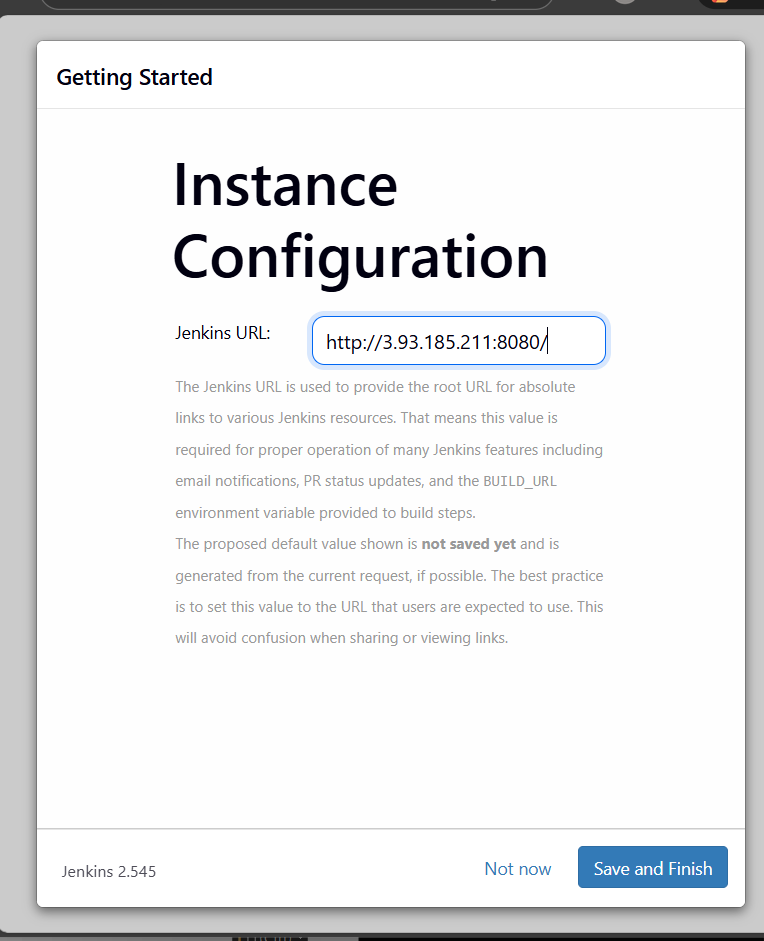
* Kabhi-kabhi browser kisi aur instance ki IP open hoti hai.  
  Confirm IP using this command
* curl ifconfig.me
* Browser me bhi wahi IP hona chahiye.

Nichay jayoh ab

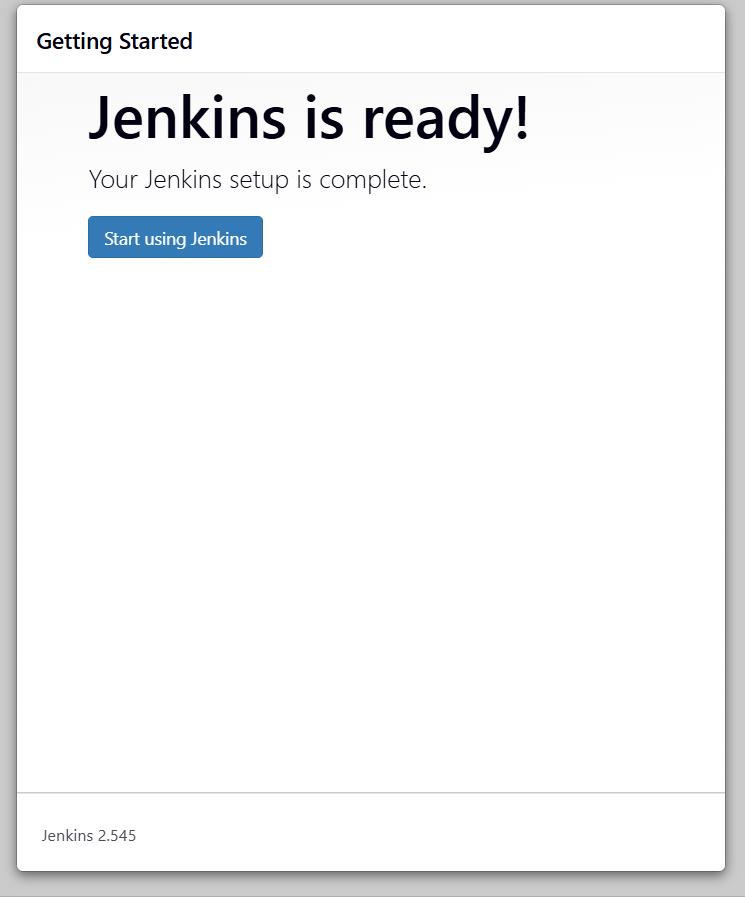








This is current Jenkins url. For ex k liye hai yeh



ab hum yeh Janengay ki jaisy kal jenkins install kiya tha usper kaam bhi kiya tha ab kal mnay sabkuch close karkay aaj dubara ssh open kiya toh aaj m kaisy vaapis jenkins ko start karu taaki kal vala bhi kaam dikha jaye

aaj Jenkins ko wapas kaisay start karny k liye use:

>> sudo snap services Jenkins

Expexted oupt: service startup current

Jenkins enabled active

Agar Jenkins band ho then use;

>> sudo snap start Jenkins

Phir dobara check:

sudo snap services jenkins

**🌐 Browser me Jenkins open karo**

<http://54.157.68.118:8080>

humay **kal wala Jenkins dashboard** + **saari jobs** dikheni chahiye 🎉

**🔒 Jenkins data kahan hota hai (snap)**

Kal ka kaam is location me saved hai:

/var/snap/jenkins/common/jenkins

**⚙️ Auto-start ensure karo (IMPORTANT)**

Usually snap by default enable hota hai, phir bhi confirm:

>> sudo snap services jenkins

Agar disabled ho to:

>> sudo snap enable jenkins

Bilkul 👍 agar **direct Jenkins dashboard** kholna hai, to bas ye cheezein ensure karo aur open karo 👇

**🌐 Direct Jenkins Dashboard ka URL**

Browser (Chrome/Edge) me ye likho:

http://54.157.68.118:8080(elastic-ip:8080)

👉 Ye **direct Jenkins dashboard** hai.

**❗ Agar page open NA ho**

**1️⃣ Jenkins running hai ya nahi**

SSH me:

sudo snap services jenkins

Status hona chahiye:

jenkins enabled active

**2️⃣ Security Group me port 8080 open hai ya nahi (MOST IMPORTANT)**

EC2 → Instances → **Security Groups → Inbound rules**

Add karo agar nahi hai:

| **Type** | **Port** | **Source** |
| --- | --- | --- |
| Custom TCP | 8080 | 0.0.0.0/0 |

👉 Save rules

**3️⃣ Jenkins ka port confirm karo**

sudo snap get jenkins http.port

Default hota hai:

8080

**🔐 First time login ho to**

Agar login page aaye:

sudo cat /var/snap/jenkins/common/secrets/initialAdminPassword

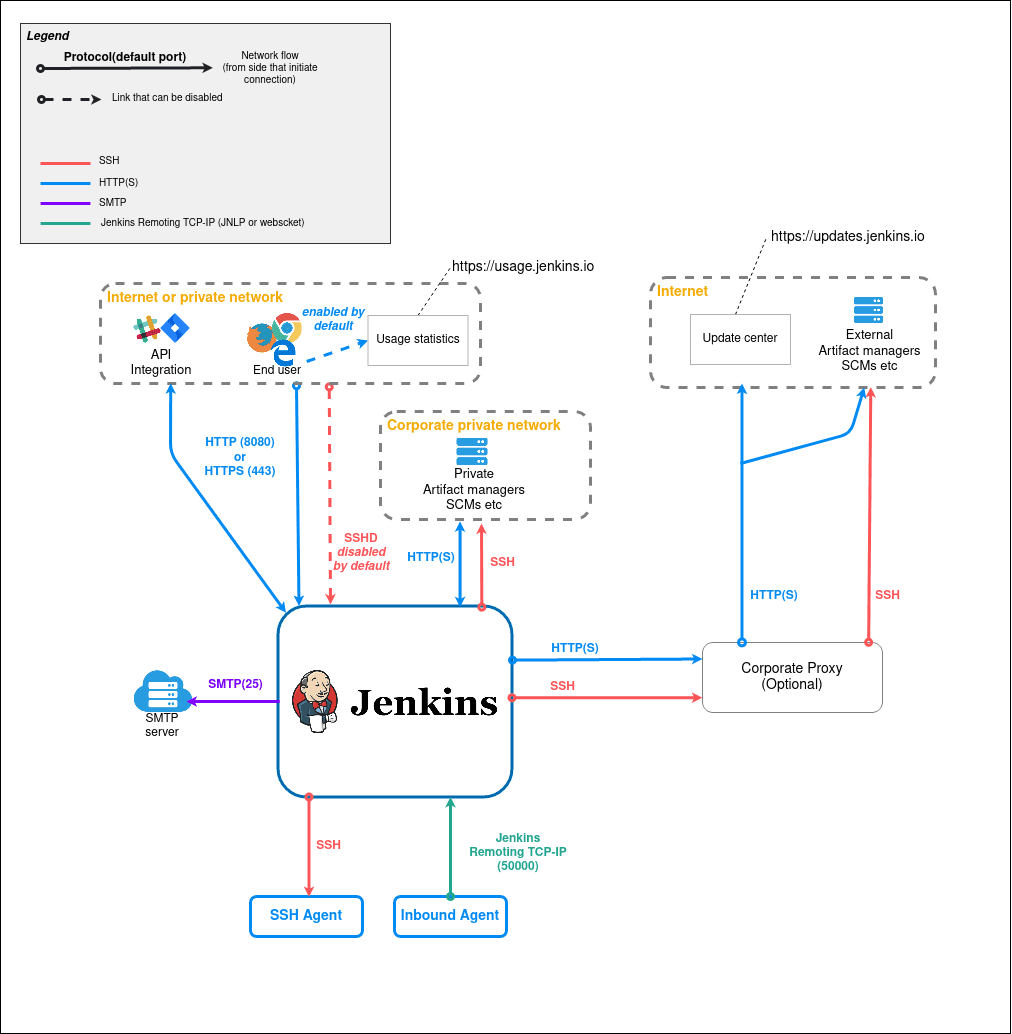
Password copy karo → browser me paste

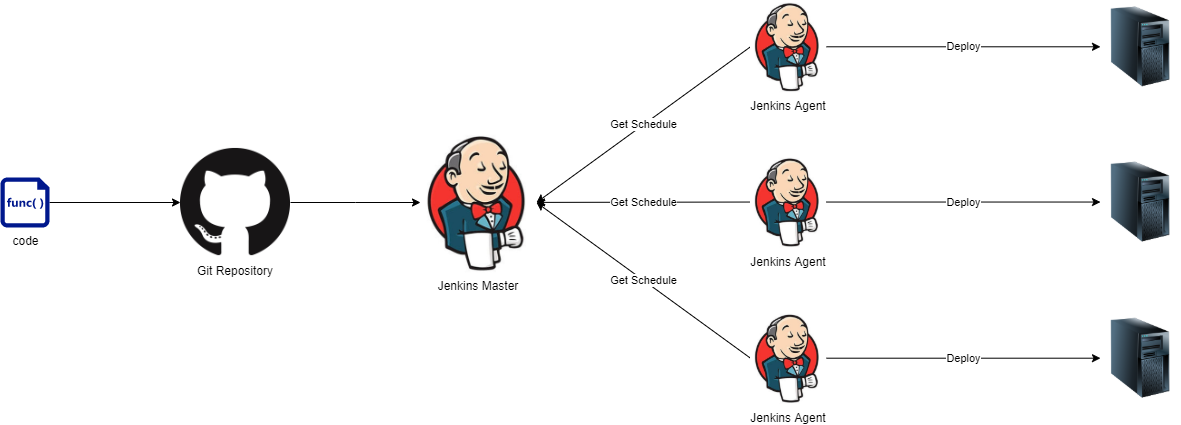
Day-3 4feb 2026

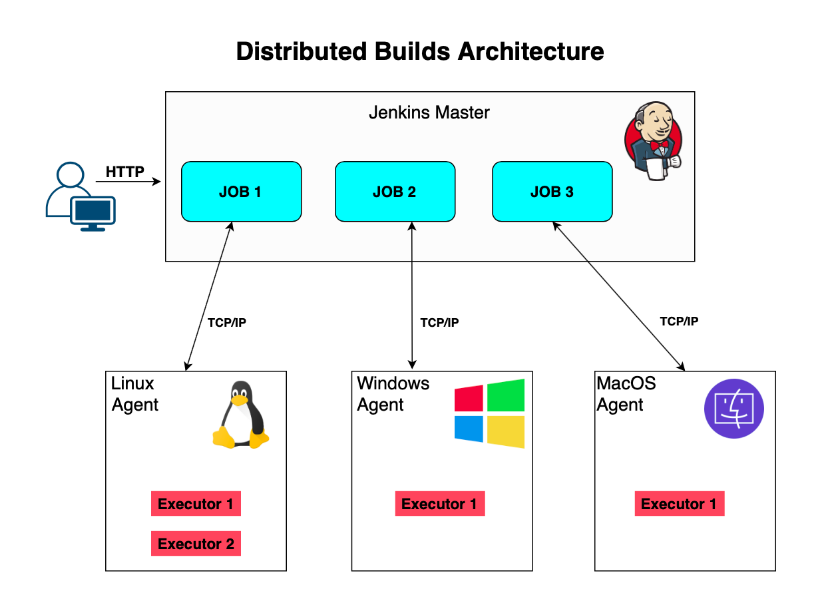
Jenkins Architecture

Jenkins Architecture batata hai ki **Jenkins internally kaise kaam karta hai**, kaun-kaun se components hote hain aur unka role kya hota hai.

Jenkins ka architecture **Master–Agent model** par based hota hai.







**Nichay jayoh**

**1️⃣ Jenkins Master**

**🔹 Master kya hota hai?**

Jenkins Master **central controller** hota hai jo Jenkins ke sabhi kaamon ko manage karta hai.

**🔹 Master ke main kaam:**

* Jenkins UI (Dashboard) provide karta hai
* Jobs / Pipelines create & manage karta hai
* Build schedule karta hai
* Agents ko instructions bhejta hai
* Plugins manage karta hai
* User authentication & authorization

**🔹 Master kya nahi karta? (Best Practice)**

* Heavy build / testing ka kaam khud nahi karta  
  👉 Ye kaam **Agents** ko deta hai (performance ke liye)

**Now we learn Jenkins agent**

**2️⃣ Jenkins Agent (Node / Slave)**

**🔹 Agent kya hota hai?**

Agent ek **machine / server** hota hai jo Jenkins Master ke kehne par **build, test, deploy** ka kaam karta hai.

**🔹 Agent ke types:**

* Permanent Agent
* Cloud Agent (AWS EC2, Docker)
* Temporary Agent

**🔹 Agent ke kaam:**

* Source code checkout
* Compile / build
* Test cases run
* Deployment

👉 Ek Jenkins Master ke **multiple Agents** ho sakte hain.

**We learn Jenkins home directory**

**3️⃣ Jenkins Home Directory**

**🔹 Jenkins Home Directory kya hoti hai?**

Ye Jenkins ka **data store** hota hai jahan Jenkins apna sara important data save karta hai.

**🔹 Jenkins Home me kya hota hai?**

* Job configuration
* Build history
* Logs
* Plugins
* User data
* Workspace

**🔹 Common paths:**

* **Linux**:
* /var/lib/jenkins
* **Windows**:
* C:\ProgramData\Jenkins\.jenkins

⚠️ Jenkins backup ka matlab = **Jenkins Home ka backup**

**4️⃣ Jenkins Port (8080)**

**🔹 Jenkins port kya hota hai?**

Jenkins by default **port 8080** par run karta hai.

**🔹 Jenkins access ka URL:**

http://localhost:8080

Ya server IP ke saath:

http://<server-public-ip>:8080

**🔹 Port 8080 kyun?**

* Non-privileged port
* Commonly used for web apps
* Easy to configure

👉 Port change bhi kiya ja sakta hai (config file se)

**🛠 Practice (Hands-on)**

**✅ Step 1: Jenkins Dashboard open karo**

Browser me:

<http://localhost(public-ip-current-instance-jiska> use karkayjenkins vs code m install kiya tha):8080

**✅ Step 2: Nodes (Agents) check karo**

Jenkins Dashboard →

Manage Jenkins → Nodes

Yahan tum dekhoge:

* Built-in Node (Master)
* Connected / Disconnected agents

**✅ Step 3: System Info check karo**

Manage Jenkins → System Information

Yahan dikhega:

* OS details
* Java version
* Memory
* Environment variables
* Jenkins Home path

**🎯 Goal – Achieved?**

Ab tum samajh paoge:

* Jenkins Master kya karta hai
* Agent ka role kya hai
* Jenkins data kahan store hota hai , Jenkins ka web access kaise hota hai and Jenkins ka web access kaise hota hai

Day-4

Jenkins-jobs(freestyle project)

**1️⃣ Jenkins Job kya hota hai?**

**Ans:**

**Jenkins Job** ek task hota hai jo Jenkins automatically perform karta hai, jaise:

* Code build karna
* Test chalana
* Script run karna
* Deployment karna

👉 Job = **automation ka unit**

**🔹 Real-life example**

Jaise alarm lagana ⏰

* Time aaya → alarm baja  
  Waise hi:
* Trigger aaya → Jenkins job chali

**2️⃣ Freestyle Project kya hota hai?**

**🔹 Freestyle Project**

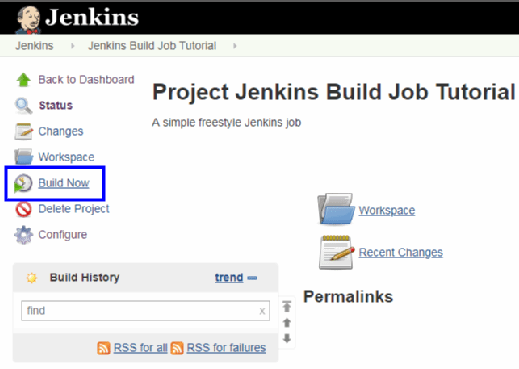
Freestyle Project Jenkins ka **sabse basic aur simple job type** hota hai.

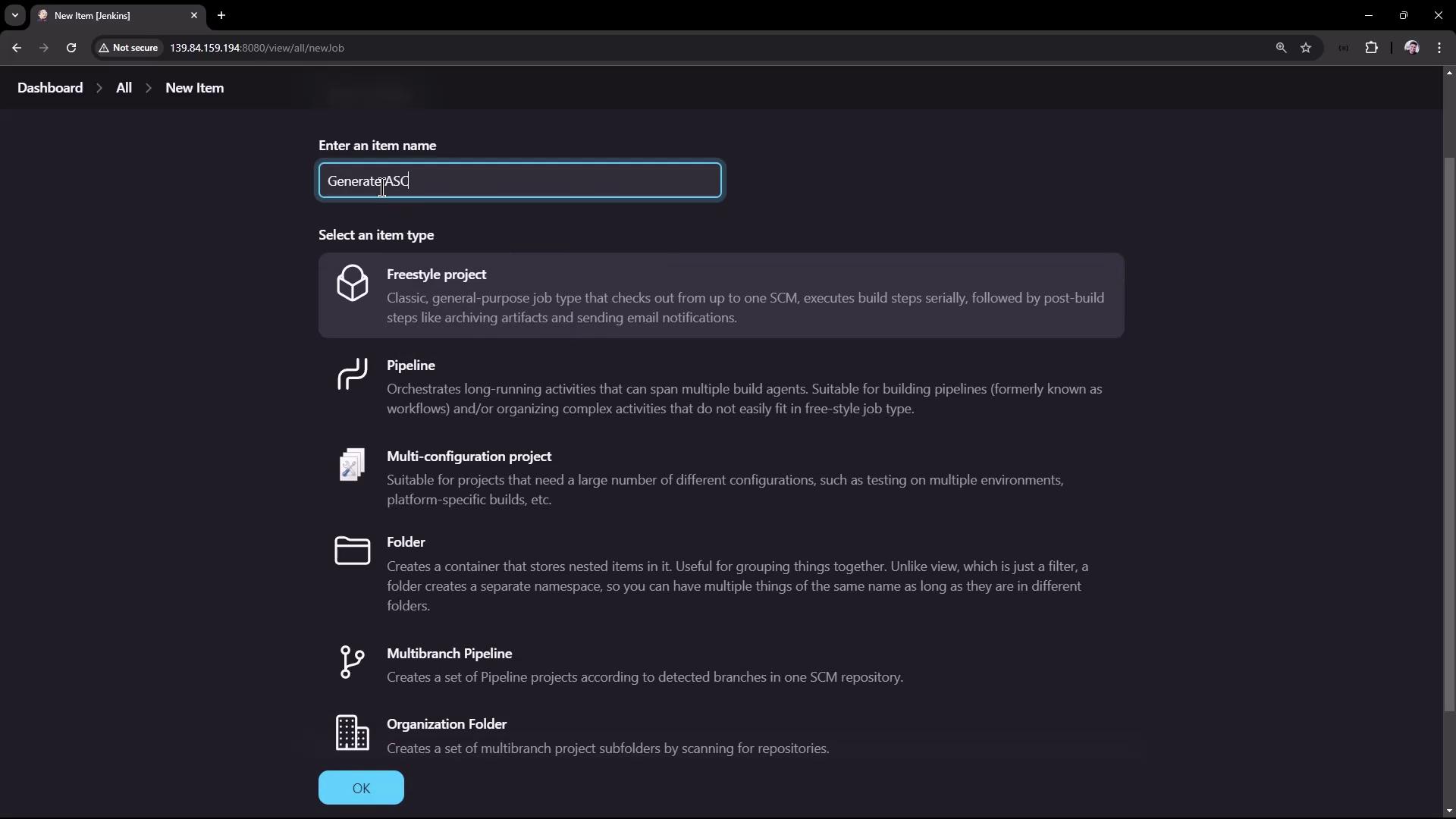
**🔹 Iska use kab hota hai?**

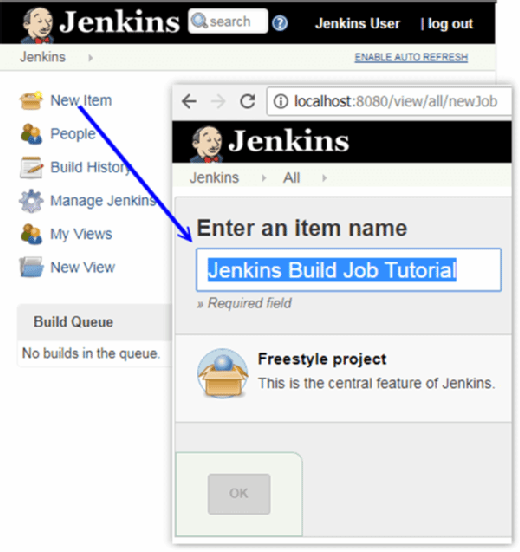
* Beginners ke liye
* Simple shell / batch command run karni ho
* Jenkins samajhne ke liye

**🔹 Freestyle Project me kya hota hai?**

* Source code (optional)
* Build triggers
* Build steps
* Post-build actions







**3️⃣ Build Triggers**

**🔹 Build Trigger kya hota hai?**

Build Trigger batata hai **job kab chalegi**.

**🔹 Common Build Triggers:**

* **Build periodically** → Time ke hisaab se
* **Poll SCM** → Git me change aaye to
* **Build manually** → “Build Now” button se

👉 Abhi practice ke liye:  
✔️ **No trigger** (manually run karenge)

**4️⃣ Build Steps**

**🔹 Build Steps kya hote hain?**

Build Steps wo commands hoti hain jo Jenkins job run karte time execute karta hai.

**🔹 Examples:**

* Shell command (Linux)
* Windows batch command
* Maven / Gradle command
* 👉 Aaj hum simple command chalayenge:

echo "Hello Jenkins"

**🛠 Practice (Hands-on)**

**✅ Step 1: Jenkins open**

Browser:

http://<ELASTIC\_IP>:8080

**✅ Step 2: New Freestyle Job banao**

1. **New Item** click karo
2. Job name:

hello-jenkins-job

1. **Freestyle project** select karo
2. **OK**

**✅ Step 3: Build Step add karo**

Scroll → **Build** section(bulid step per click)

**🔹 if we select**

**Linux / Ubuntu Jenkins:**

* **Add build step → Execute shell**
* Command:

echo "Hello Jenkins"

**🔹if we select**

**Windows Jenkins:**

* **Add build step → Execute Windows batch command**
* Command:

echo Hello Jenkins

👉 **Save**

**✅ Step 4: Job run karo**

* Job page → **Build Now** 🔨

**✅ Step 5: Output check karo**

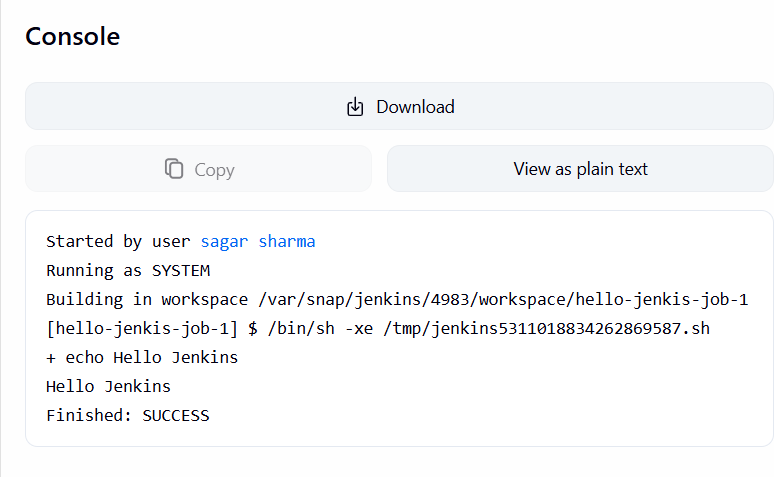
* Build number (#1) → **Console Output**

Expected:

Hello Jenkins

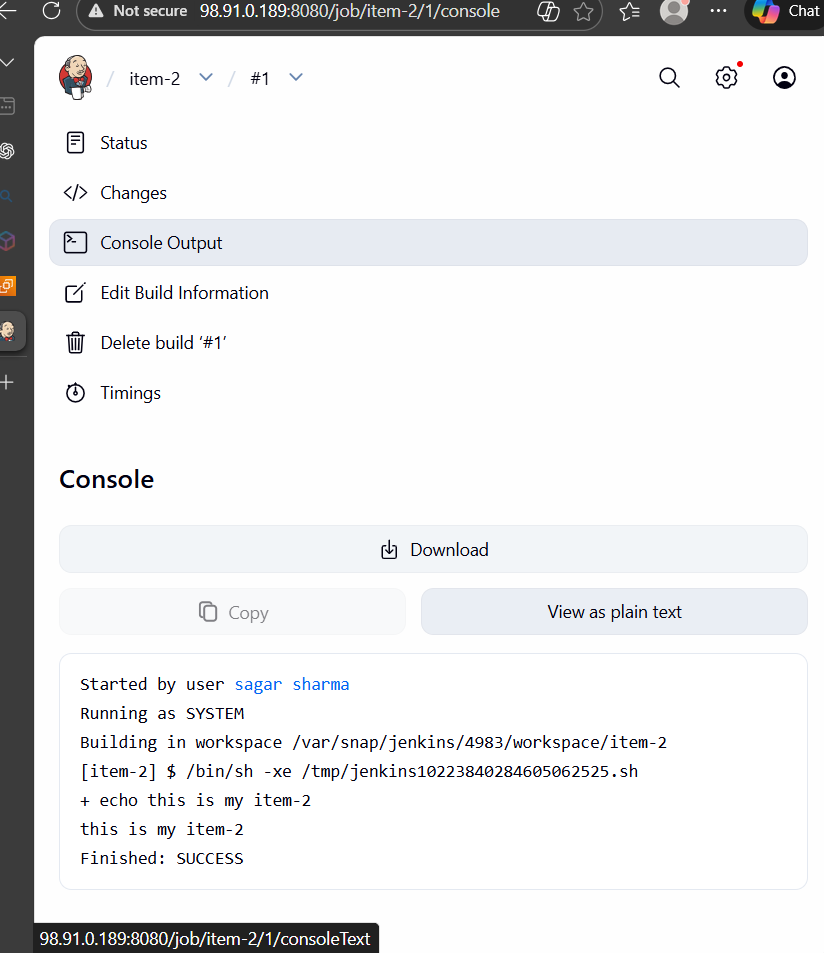
Finished: SUCCESS

🎉 Job successfully run!



Now we make new item which is item-2 then add bulid step then execute shell windows Jenkins slect karkay yeh likhdoh

echo is my item-2 then save then build-now green-color-number per click then console output



****

**🎯 Goal – Achieved ✅**

Ab tum:

* Jenkins job kya hoti hai ✔️
* Freestyle project create kar sakte ho ✔️
* Build step add karke command run kar sakte ho ✔️

Day-5

Jenkins + Git Integration

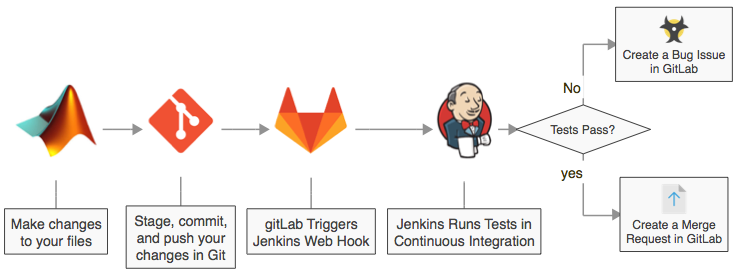
1️⃣ Git + Jenkins Integration kya hota hai?

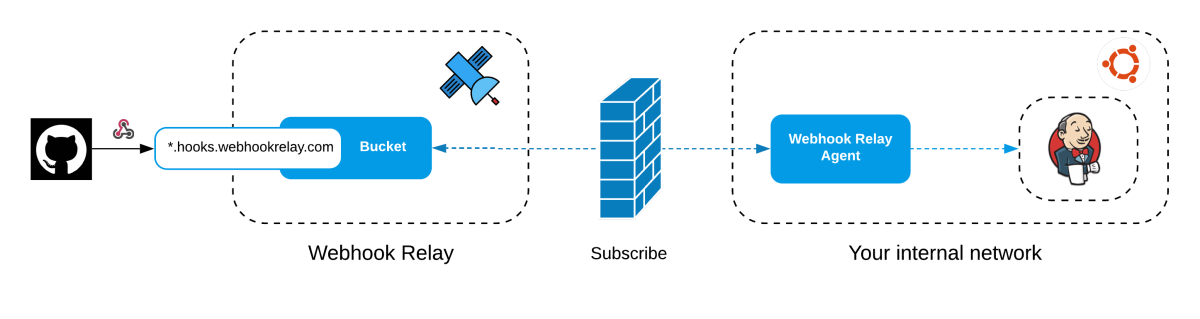
👉 Git + Jenkins integration ka matlab:

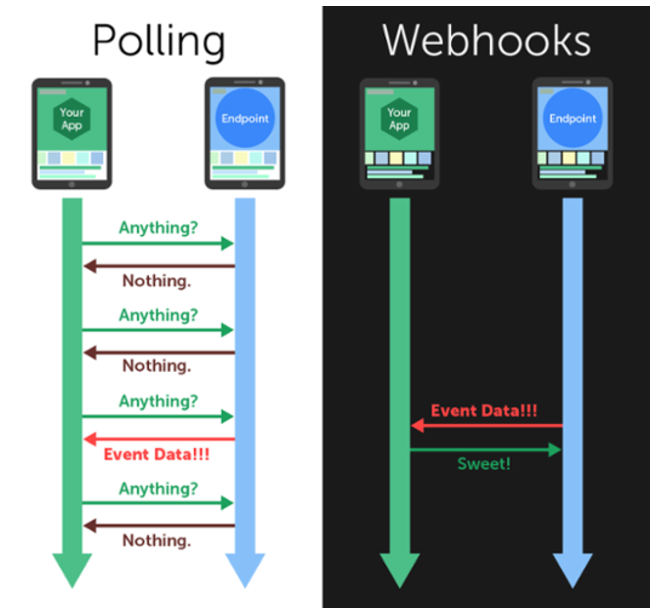
* Code GitHub repo me hai
* Jenkins us repo se code pull karta hai
* Aur automatically build chalata hai

Simple flow (yaad rakhna):

Developer → Git push → Jenkins → Build/Test







2️⃣ Jenkins GitHub se code kaise pull karta hai?

Jenkins job me:

* Source Code Management → Git
* Repo URL diya jata hai
* Jenkins internally:

git clone / git pull chalata hai

Poll SCM vs GitHub Webhook

🔹 Poll SCM(source code management)

* Jenkins ko bar-bar GitHub check karta hai
* “Code change hua ya nahi?”

⏰ Example:

\*/5 \* \* \* \*

👉 Har 5 minute me check

✅ Easy  
❌ Time waste + delay

🔹but GitHub Webhook (BEST)

* GitHub khud Jenkins ko batata hai
* Jaise hi git push hota hai

⚡ Instant build  
✅ Industry standard  
❌ Thoda setup chahiye

🛠 Practically implement:

🎯 Target:

👉 GitHub repo connect ho  
👉 Jenkins automatically code pull kare

Practically implementation:

**✅ GitHub par commit aur push kaise karein (Step-by-Step)**

**🔹 Step 1: GitHub par new repo banao**

1. GitHub open karo
2. **New repository**
3. Repo name:

jenkins-git-demo(kuch bhi rakh doh)

1. **Public** select karo
2. **Create repository**

👉 Ab GitHub tumhe repo ka URL dega, jaise profile p jakay repo per click karengay ab jo repo bnayi thi voh dikhega usko scroll karengay tog url dikh jayega is type may:

<https://github.com/github-username/jenkins-git-demo.git>

this-is-our-github-username(sagarvashist19-boop)

ex:

<https://github.com/sagarvashist19-boop/jenkins-git-demo.git>

**🔹 Step 2: Apne system par folder banao**

Terminal / Git Bash / VS Code terminal me yeh command chalyoh:

>>mkdir jenkins-git-demo : (mkdir k baad voh daaloh jis naam say repo banyi):

>> yeh command Jenkins name ka folder bna degi.

>>cd jenkins-git-demo :

(Ismay bhi cd m reponame) :

>> yeh command current folder say Jenkins folder per shift kardegi

**🔹 Step 3: Git initialize karo**

git init

Output aayega:

Initialized empty Git repository

**🔹 Step 4: File banao (hello.txt) in vs terminal;**

echo "Hello from GitHub" > hello.txt

Check karo: type ls

**Step 5: File ko staging me daalo**

git add hello.txt ya git add .(yeh better hoga)

**🔹 Step 6: Commit karo**

git commit -m "first commit for jenkins"

Output:

1 file changed, 1 insertion(+)

**🔹 Step 7: GitHub repo ko remote add karo**

⚠️ USERNAME apna GitHub username rakho yeh command in vs terminal

git remote add origin <https://github.com/USERNAME/jenkins-git-demo.git>

exx:

<https://github.com/sagarvashist19-boop/jenkins-git-demo.git>

Verify: git remote -v

Yeh command hmaray outside servers url dekhta hai in

Which our local repository linked.

**🔹 Step 8: Push karo (IMPORTANT)**

git branch -M main

git push -u origin main

inkay baad yeh humsay username for github and password puchega password k andar hum PATfill karengay .

No we make PAT(personal access token) :

**🔐 GitHub PAT kaise banayein:**

1. GitHub → **Settings**
2. **Developer settings**
3. **Personal access tokens**
4. **Tokens (classic)** → **Generate new token**

**5. New token page par kya-kya select karna hai**

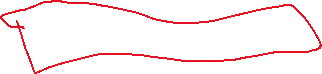
**Note (optional)**

jenkins-token

**Expiration**

1. 30 days / 90 days / custom (jo chaho)agar 90 days tak ka choose kiya toh 90 days taky ahi chalega .
2. Select scopes:
   * ✅ repo
3. **Generate token**
4. **Token COPY** kar lo (dobara nahi dikhega)
5. **Bottom par click karo Generate token**
6. github\_pat\_11BWHHR3Y07wqfDfF7Zalo\_kLQwq8GGYZNYdW9tma5ByK4SCxwYksqkN3IuOYw8XFzWU2FBYXFMs5UVVAY
7. ssh -T git@github.com





Yeh red box hmara PAT hai isay copy karkay jo password puchra hai vs code terminal github-username fill karnay k baad usmay paste kardoh

**✅ Successful push ka sign:**

Terminal me dikhega:

Enumerating objects...

Writing objects...

\* [new branch] main -> main

Aur GitHub repo refresh karoge →  
hello.txt dikh jayegi 🎉

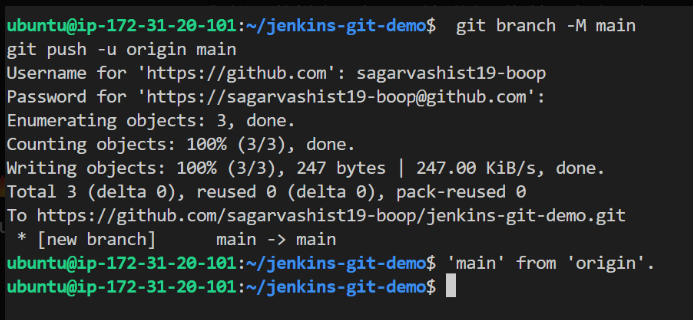
**🎯 Ab hum yahan tak aa chuke hai:**

✔️ Correct remote URL  
✔️ Correct branch (main)  
✔️ Authentication step reached

👉 Bas **username + PAT** dalna hai.

**Jab push complete ho jaye, bas likho:**

**“Git push successful”**



Final step karengay github and Jenkins integeration.

🚀 Next step: Jenkins + GitHub Integration (FINAL PART):

✅ Ab jo setup tumne kiya hai — ✔️ CORRECT

1️⃣ Source Code Management (Git)

* Repository URL
* https://github.com/sagarvashist19-boop/jenkins-git-demo-2.git
* Credentials: GitHub token ✅ (public repo hai, phir bhi ok)
* Branch Specifier
* \*/main

✔️ Sahi hai (ab repo me main branch + commit exist karta hai)

2️⃣ Repository me file bhi aa chuki hai ✅

Tumhare GitHub screenshot me clearly dikh raha hai:

hello.txt

Content:

Hello Jenkins

This is my first CI test.

✔️ Matlab repo ab empty nahi hai  
✔️ main branch me commit exist karta hai

3️⃣ Build Steps → Execute shell

Tumne jo likha hai:

ls

cat hello.txt

✔️ PERFECT  
✔️ ls = files dikhaega  
✔️ cat hello.txt = file ka content print karega

🚀 Ab NEXT STEP (IMPORTANT)

👉 Bas yeh karo:

1. Save button dabao
2. Job ke main page par jao
3. Build Now click karo ▶️

🔍 Expected Output (Console Output me)

Agar sab sahi hua (aur ab hua hi hai 😄), to console me kuch aisa dikhega:

hello.txt

Hello Jenkins

This is my first CI test.

Aur upar: Finished: SUCCESS 🎉🎉🎉

❌ Pehle “revision not found” error KYUN aa raha tha?

Short & clear reason 👇

❌ Pehle:

* Repo empty thi
* main branch me koi commit hi nahi tha
* Jenkins jab yeh chalata tha:
* git rev-parse origin/main

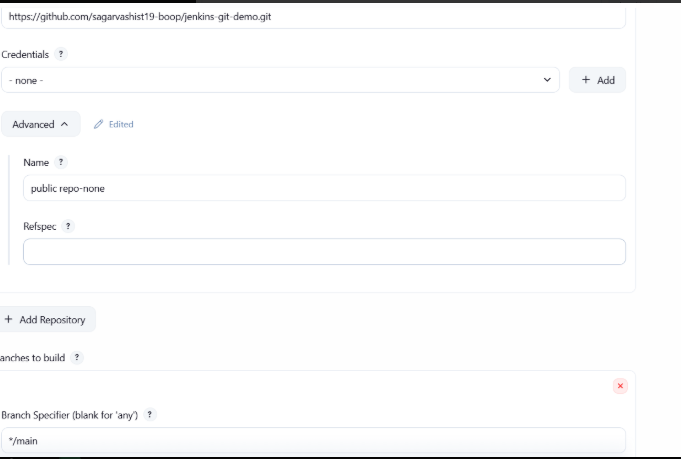
to Git bolta tha:  
❌ *“bhai main to exist hi nahi karta”*

✅ Ab:

* hello.txt add ho chuki hai
* main branch me commit hai
* Jenkins ko revision mil jaata hai
* Error khatam ✅

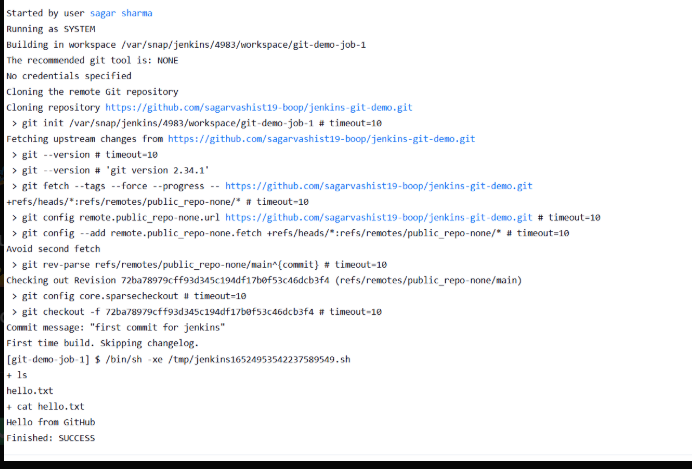
Nichay jayoh

Like this:

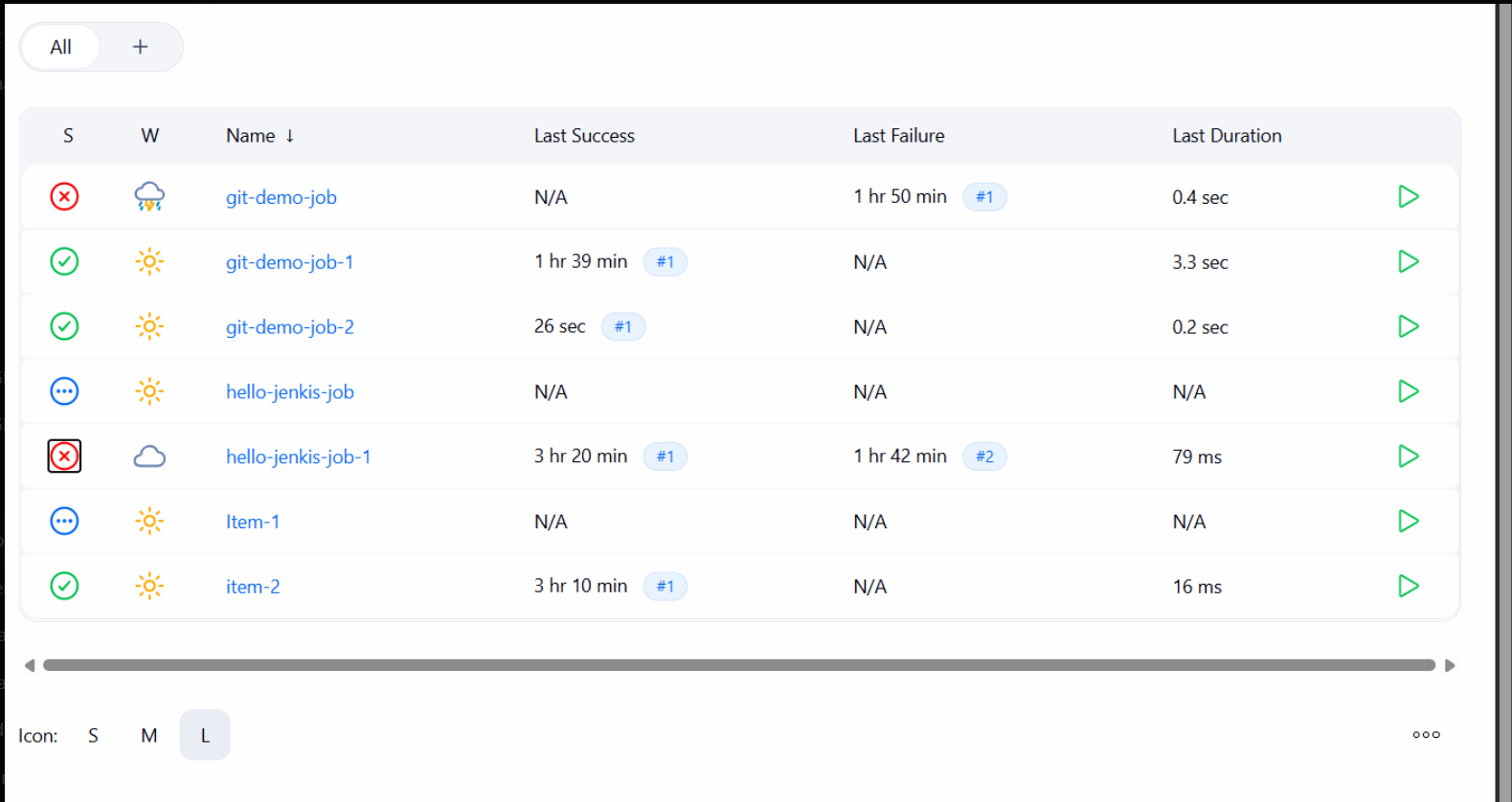


Console Output me dikhega:

Hello from GitHub finished succes

Like this: 

🎉 Jenkins ne **GitHub se code pull kar liya**



**Red cross (❌) wali Jenkins jobs delete karni hain.**  
Yeh bilkul safe aur simple hai. Step-by-step dekh lo 👇

**✅ Method 1: Jenkins UI se Job Delete (Easiest)**

**🔹 Step 1: Job ke naam par click karo**

Jaise: **git-demo-job** ya **hello-jenkis-job-1** (jis par ❌ laga hai)

**🔹 Step 2: Left sidebar me “Delete Project” par click karo**

📍 Left side menu me niche ki taraf milega

**Step 3: OK / Yes confirm karo**

Bas! Job permanently delete ho jayegi ✅

🔁 **Har red cross wali job ke liye same steps repeat karo**

**⚠️ Agar “Delete Project” option nahi dikh raha**

Toh do reasons ho sakte hain:

**🔸 1. Tum login admin se nahi ho**

👉 **Admin user** se login karo

**🔸 2. Delete option hide hai**

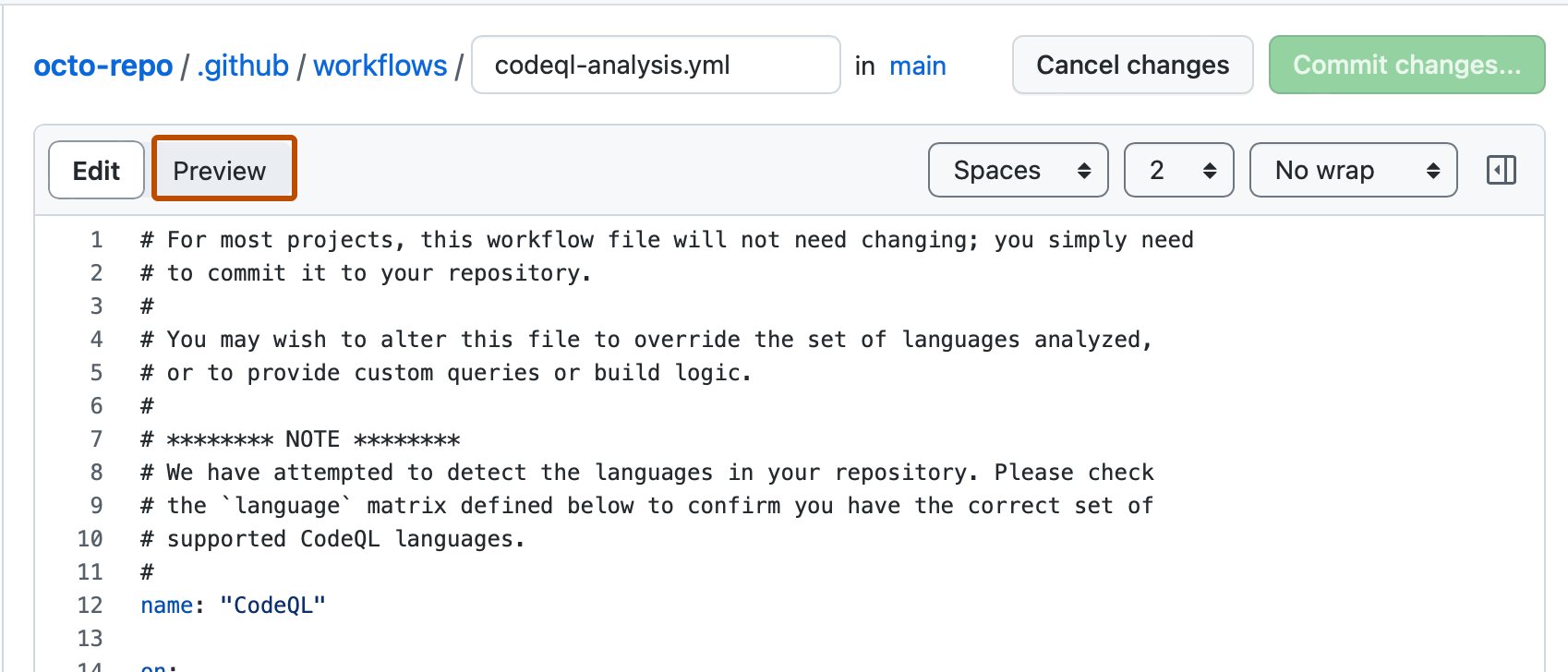
Toh ye karo:

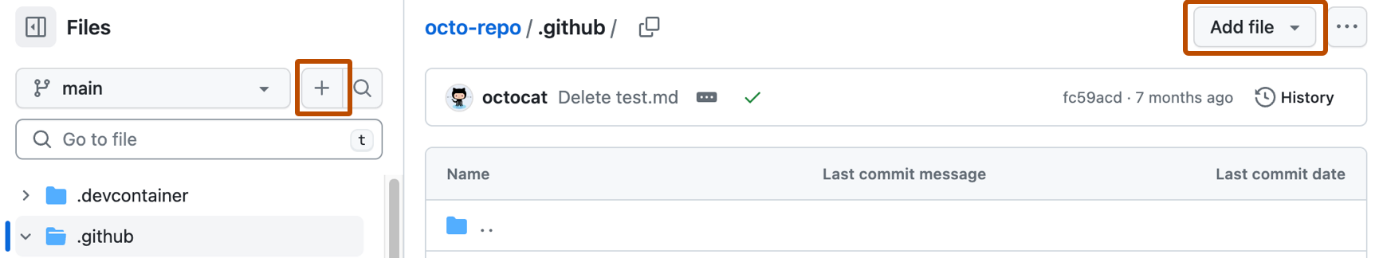
**Jenkins Dashboard → Manage Jenkins → Configure Global Security**

* “Enable project-based security” unchecked ho
* Ya admin ko full permission ho

Save karo 🔐

**GitHub repo me new file add karne ke 2 easy methods** hote hain — **browser se** aur **VS Code/Git se**. Pehle **browser wala (beginner-friendly)** bata raha hoon.







**🟢 METHOD 1: GitHub WEBSITE se (NO coding, easiest)**

**STEP 1️⃣ Repo open karo**

👉 GitHub pe login  
👉 Apna **repository** open karo

**STEP 2️⃣ Add file pe click karo**

Repo ke right side:

Add file → Create new file

**STEP 3️⃣ File ka naam likho**

Upar box me filename likho, example:

README.md

index.html

main.java

📌 Folder ke andar chahiye ho toh:

src/Main.java

**STEP 4️⃣ File ka content likho**

Neeche editor me code ya text likh do ✍️

Example:

Hello GitHub

This is my first file

**STEP 5️⃣ Commit new file**

Neeche scroll karo:

* Commit message likho (example: Added README file)
* **Commit new file** button dabao ✅

🎉 DONE — file repo me add ho gayi

**🟡 METHOD 2: VS Code / Git se (DevOps way)**

**STEP 1️⃣ Local repo me file banao**

touch test.txt

**STEP 2️⃣ Git status check**

git status

**STEP 3️⃣ File add karo**

git add test.txt

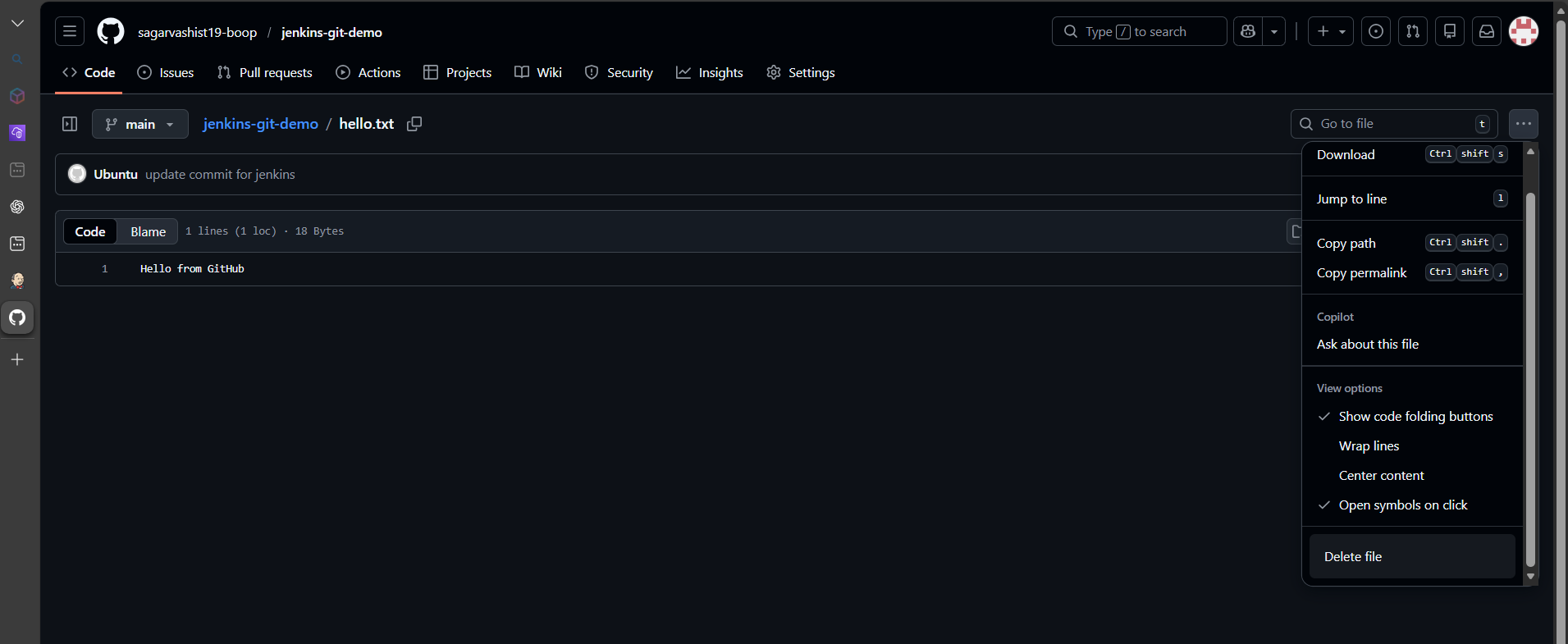
**STEP 4️⃣ Commit**

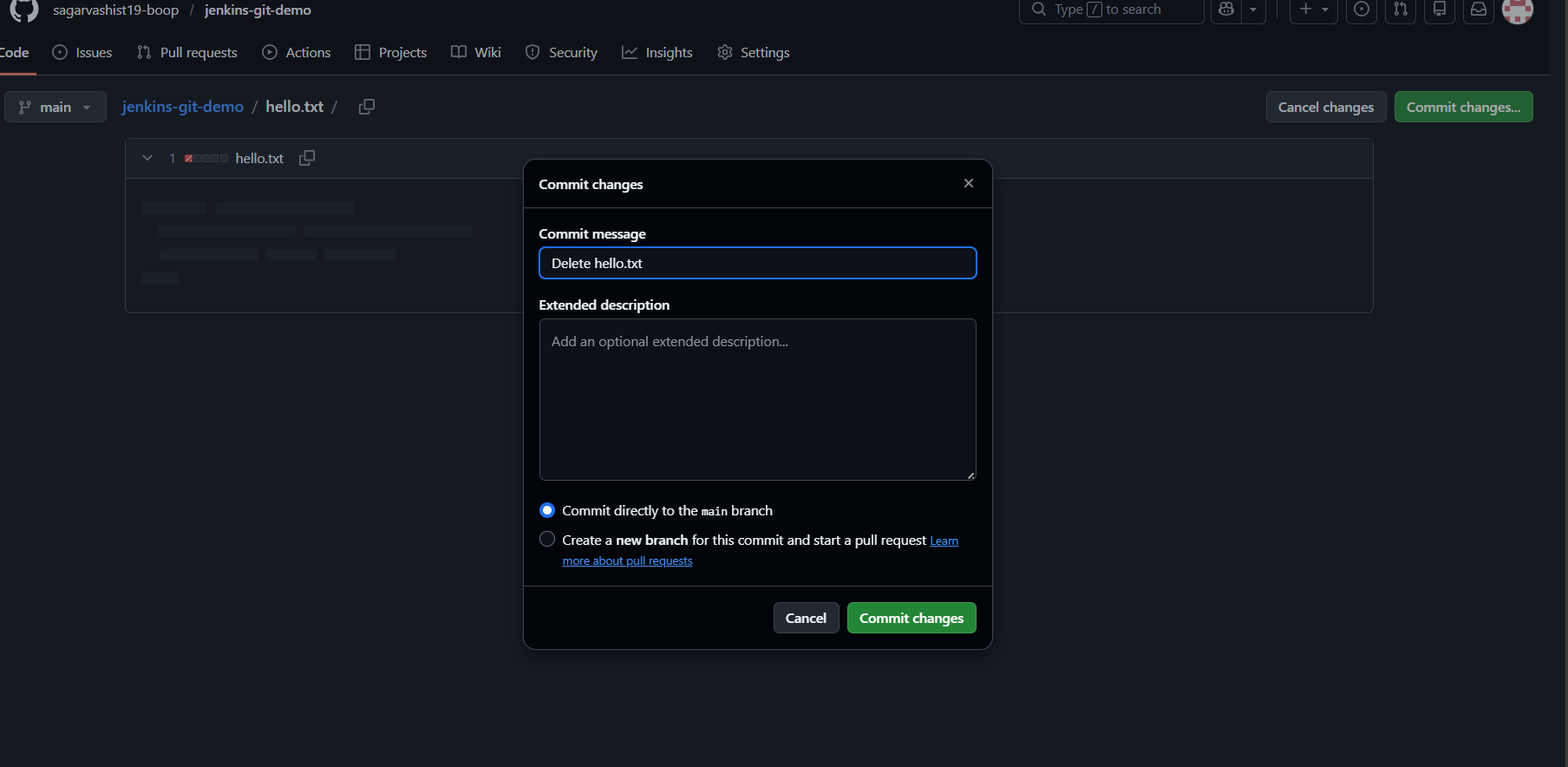
git commit -m "Added test.txt file"

**STEP 5️⃣ GitHub pe push**

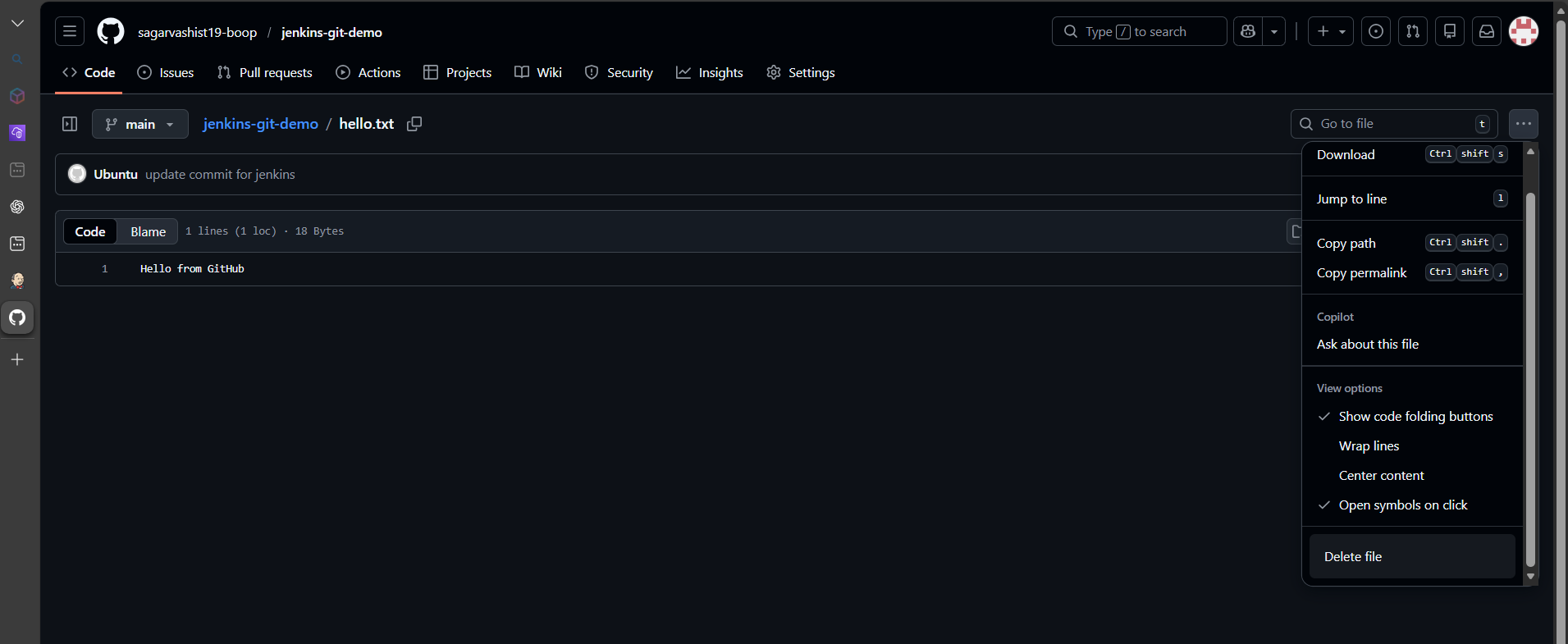
git push origin main

agar hum kisi file ko repo say delete karna chathay hai then us repo ko open karoh jo file delete karni hai . file p click kartay hai them 3 dots per click karkay scroll down karoh then delete





Agar humay kio bhi repo delte karni hai toh us repo pr click karoh then jo code issue vaali line haii ismay settings hai us per click karo then scroll down danger zone then type repo name which you want to delte then repo will delete.



**✅ Agar Jenkins dashboard direct khul raha hai…**

👉 **Tum Jenkins me kaam kar sakte ho**  
👉 **Jobs bana sakte ho, run kar sakte ho, delete bhi**

**Matlab:**

* Login page nahi aa raha
* Username/password nahi pooch raha  
  ➡️ Jenkins **unsecured / open mode** me chal raha hai

Ye **learning / lab environment** me normal hai 👍

**🧠 Aisa kyun hota hai?**

Usually jab:

* Jenkins fresh install hota hai
* Security enable nahi hoti
* Ya internal / local server hota hai

Tab Jenkins **direct dashboard open** kar deta hai

**🔍 hum kya-kya kar sakte hai?**

Agar dashboard open hai aur ye options dikh rahe hain:

* ➕ **New Item**
* ⚙️ **Manage Jenkins**
* 🗑️ **Delete Project**
* ▶️ **Build Now**

👉 Matlab tum **ADMIN ho** 💪  
👉 Tum **full control** me ho

**❌ Kab kaam nahi kar paoge?**

Sirf tab jab:

* Dashboard open ho
* **New Item / Configure / Delete** disabled ho
* Ya “Permission denied” aaye

Tab issue hota hai **role / permission ka**

**🔐 Production me kya hota hai?**

Real company setup me:

* Login compulsory hota hai
* Roles hote hain (Admin / Developer / Viewer)
* Direct open dashboard ❌ allowed nahi

But **learning ke liye jo tum kar rahe ho, wo 100% sahi hai** 👍

Day-6

Jenkins + Maven

(build—automation)

**🟡 Maven Basics**

**🔹 Maven kya hai?**

👉 **Maven ek build automation tool hai**  
👉 Java project ko **compile, test, package (JAR/WAR)** karne ke kaam aata hai

**Maven kya manage karta hai?**

* Dependencies (JAR files)
* Build process
* Project structure

**🔹 Maven Project ka main file may**

pom.xml hota hai

👉 Isme hota hai:

* Project name
* Dependencies
* Build configuration

**🟡 Build** Lifecycle **(VERY IMPORTANT):**

**Maven lifecycle ke phases:**

clean → compile → test → package → install → deploy

|  |  |
| --- | --- |
|  | **>>Kya Jenkins ke sath Java install ho gayi?**  **✅ YES (internally)**   * Jab tum Jenkins **snap se install** karte ho * Snap **apni embedded Java** use karta hai * Isliye Jenkins **chal raha hai bina issue** ke * humhare **Ubuntu system** me java command available nahi hai * Isliye terminal me java -version fail ho raha hai   📌 **Conclusion (yaad rakhna)**  Snap Jenkins works with its own Java, but system Java must be installed separately for Maven and manual builds.  **✅ STEP 1: System Java install karo (REQUIRED for Maven)**  **🔹 Command**  sudo apt install openjdk-17-jdk -y  **One-line meaning:**  Java 17 (JDK) system me install karna  **🔹 Verify Java**  java -version  **Expected Output:**  openjdk version "17.0.x"  **Output meaning:**  Java successfully installed ✔️  **✅ STEP 2: Maven install karo**  **🔹 Command**  sudo apt install maven -y  **One-line meaning:**  Maven build tool install karna  **🔹 Verify Maven**  mvn -version  **Expected Output:**  Apache Maven 3.x.x  Java version: 17  **Output meaning:**  Maven Java 17 ke sath properly linked hai ✔️ |

**Simple samjho:**

|  |  |
| --- | --- |
| clean | Purani build delete |
| compile | Java code compile |
| test | Test cases run |
| package | JAR file banata |
| install | Local repo me copy |

**✅ STEP 3: Maven project create karo**

mvn archetype:generate

Maven ka ready-made Java project banana

**Inputs (enter karo):**

groupId: com.demo

artifactId: demo-app

version: 1.0

package: com.demo

yeh output nahi ata hai toh then ctrl+c

**🟢 STEP 2: DIRECT QUICKSTART COMMAND use karo**

⚠️ Ye sabse important hai

Terminal me **exactly ye command paste karo** 👇

mvn archetype:generate \

-DgroupId=com.example \

-DartifactId=jenkinsdemo \

-DarchetypeArtifactId=maven-archetype-quickstart \

-DinteractiveMode=false

👉 **Enter** dabao

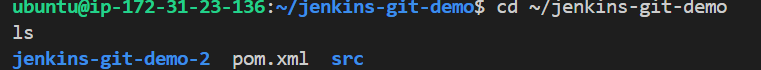
**🎯 Is command ka fayda**

* ❌ Koi list nahi aayegi
* ❌ Spark / Scala kabhi nahi aayega
* ✅ Direct **Java Maven project** banega
* ✅ Jenkins + Maven ke liye PERFECT

**🟢 STEP 3: Verify karo**

cd ~/jenkins-git-demo

ls





👉 Matlab (one line):

Sirf ek hi pom.xml hai, aur woh jenkins-git-demo folder ke andar hai.

❌ Isliye pom.xml nahi milega is folder m

Jenkins-git-demo-2

~/jenkins-git-demo/jenkins-git-demo-2/

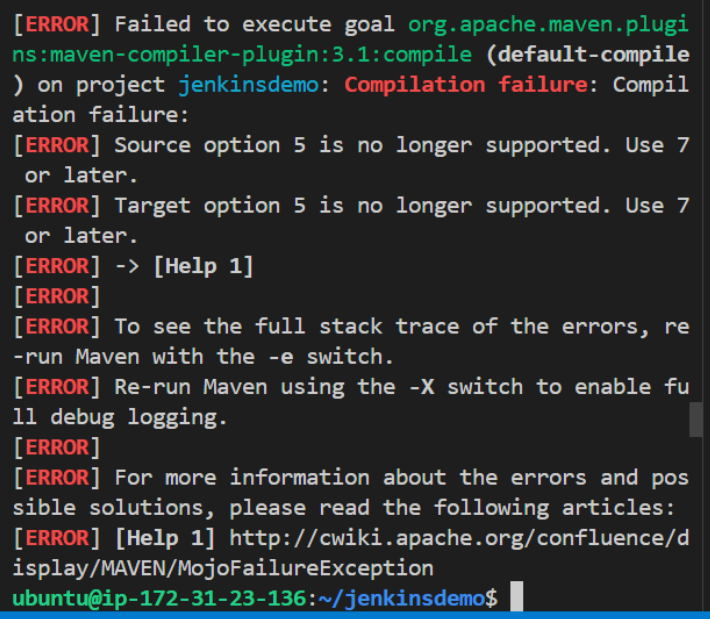
**🟢 STEP 4: Local build test**

mvn clean package

Agar aaye:

BUILD SUCCESS

But we got this



**🧠 Ye problem kyun aayi?**

maven-archetype-quickstart **default Java 5** set kar deta hai:

<source>1.5</source>

<target>1.5</target>

Lekin tum **Java 17** use kar rahe ho.

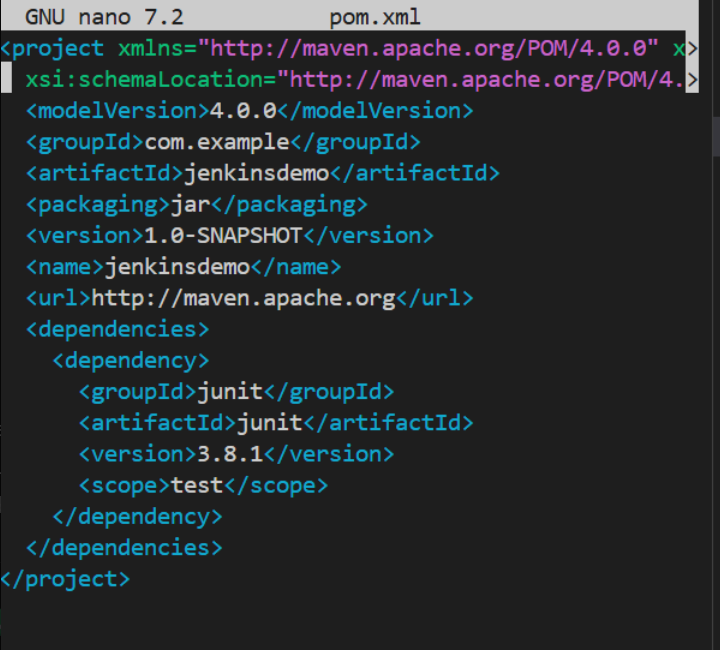
**Nichay jayoh isay fix kartay hai**

**✅ FIX this (STEP-BY-STEP):**

**🟢 STEP 1: pom.xml open karo**

nano pom.xml

ya vi pom.xml



**✅ KYA ADD KARNA HAI (IMPORTANT)**

**🔹 <dependencies> ke baad aur </project> se pehle**

ye block paste karo 👇

<properties>

<maven.compiler.source>17</maven.compiler.source>

<maven.compiler.target>17</maven.compiler.target>

</properties>

**🧩 FINAL pom.xml STRUCTURE (samajhne ke liye)**

Tumhara file end me **aisa dikhna chahiye**:

</dependencies>

<properties>

<maven.compiler.source>17</maven.compiler.source>

<maven.compiler.target>17</maven.compiler.target>

</properties>

</project>

⚠️ Dhyan rahe:

* <project> ke **andar** ho
* </project> ke **baad nahi**

**💾 SAVE & EXIT (nano editor)**

1️⃣ **CTRL + O** → Enter (save)  
2️⃣ **CTRL + X** (exit)

**▶️ AB COMMAND RUN KARO**

mvn clean package

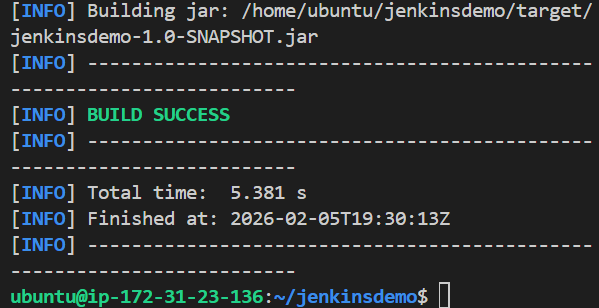
**✅ Expected result:**

BUILD SUCCESS

🎉 **Error permanently fix ho jayega**

**🧠 One-line yaad rakhna (exam + practical)**

Java version mismatch ko fix karne ke liye pom.xml me maven.compiler.source & target set karte hain.



Agar hum ab saray sab kuch close kardete hai toh humay maven ko dubara install karna nahi padega we check maven already install ya nhi using

mvn -version

**Agar Maven installed hoga to output aayega:**

Apache Maven 3.x.x

Maven home: /usr/share/maven

Java version: 17.x.x

👉 Matlab:

* Maven , java and environment hai already

Agar command not found error aye toh maven ko instll karengay using

sudo apt install maven -y

 Snap Jenkins system PATH use karta hai

 Jenkins job me mvn directly chal jaata hai

**Kya pom.xml ka key-pair se koi link hota hai?**

**✅ Simple answer:**

**Nahi.**  
pom.xml ka **SSH key / key-pair / credentials** se **koi direct relation nahi hota** ❌

Now we make maven freestyle project without git:

Click on item- make anye name

Ex Jenkins-maven-freestyle-project-1

Then click on freestyle then ok

Them add build step

Select

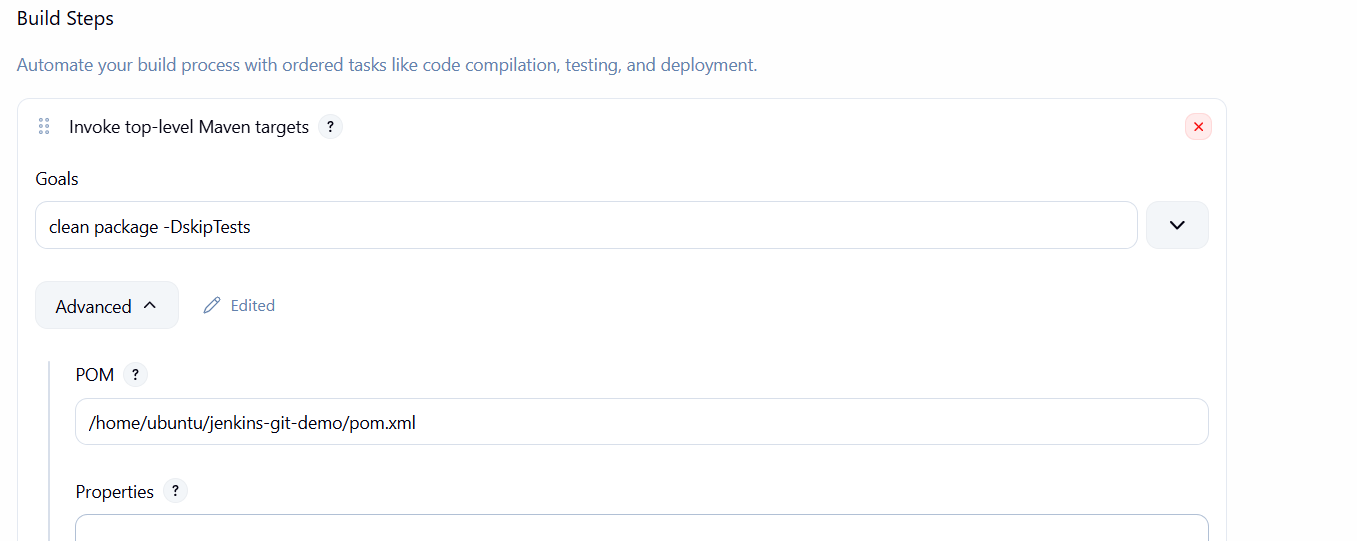
invoke top-level Maven targets

goals: clean package -DskipTests

advanced pom;

/home/ubuntu/jenkins-git-demo/pom.xml

Like this



Now we make Jenkins may maven+git project:

Click on item:

Then freestyle then save

None nahi karna hai git ko select karna then

Repo url:

<https://github.com/sagarvashist19-boop/jenkins-git-demo-2.git>

Credentials: Github token

Branch Specifier: \*/main

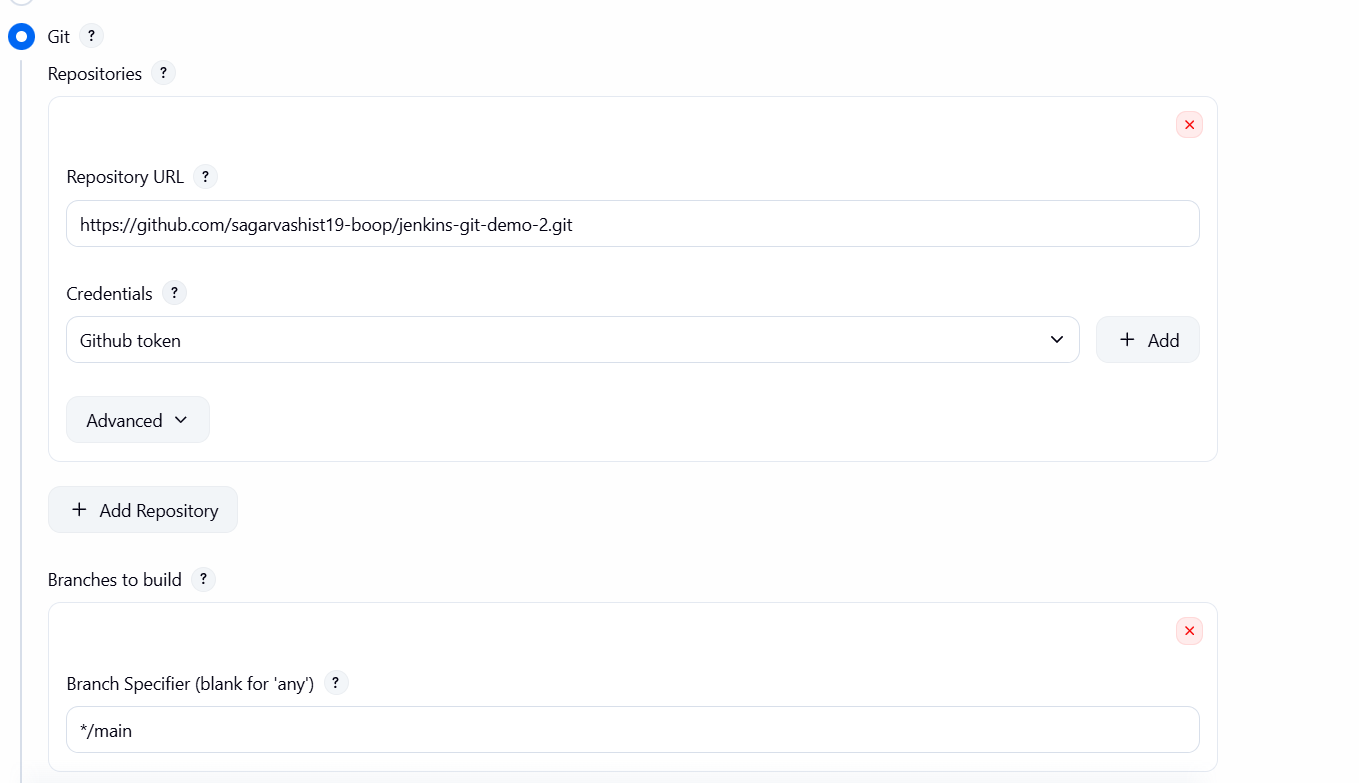
Build Steps:

select karoh

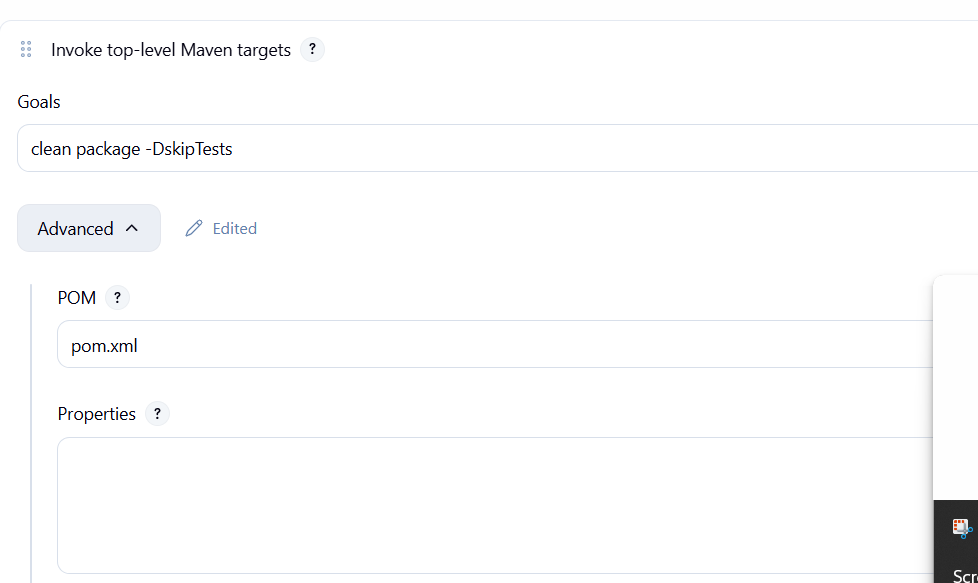
Invoke top-level Maven targets

Goals : clean package -DskipTests

Advanced POM: pom.xml



>>>

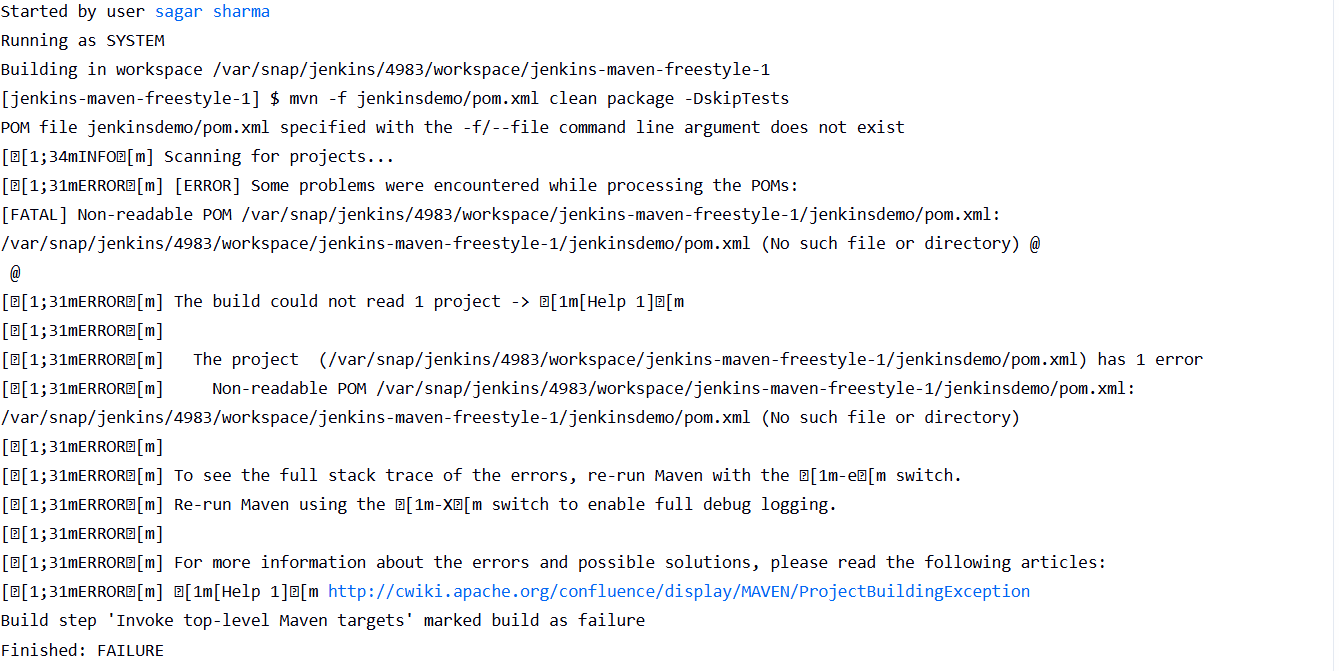


Then save then build now

Then success



Agar hum advanced pom and branch specifir nahi likhengay toh error ayega buld fail hojayega



**⚠️ Ek last IMPORTANT check (Snap Jenkins users ke liye):**

Jenkins **snap sandbox** me chalta hai. Kabhi-kabhi /home/ubuntu access blocked hota hai.  
Agar build fail ho aur **permission error** aaye, to ye ek command chala do (one-time):

sudo snap connect jenkins:home

**One-line meaning:**

Jenkins snap ko /home directory access dena

(Only tab zaroori hai jab permission error aaye.)

**Step 4: Save → Build Now ▶️**

Agar goals may clean package likhtay clean package -DskipTests hki jagah bohot time leta hau  
**REAL reason**

**1️⃣ reason First-time Maven build hai**

Jab Jenkins **pehli baar** mvn clean package chalata hai, to Maven:

* Central repository se **dependencies download** karta hai

Junit ,maven plugins ,aur kuch core jars download hotay

📦 Ye sab **internet se** aata hai  
➡️ **1–5 minutes** lagna normal hai (kabhi-kabhi zyada bhi)

**2️⃣ reason Jenkins SNAP + permissions:**

Tum **Snap Jenkins** use kar rahe ho aur **external path**:

/home/ubuntu/jenkinsdemo/pom.xml

Jenkins ko:

* /home access
* file read permission

ye sab check karna padta hai  
➡️ thoda overhead aata hai

**🔔 Agar hum abhi kuch nahi badalte hai**

* Aur wait karoge → **indefinite time** (clear end nahi)
* Learning ke liye **worth it nahi**

🎯 Goal achived:

👉 Java project ka automatic build

**Day 7 – 7feb 2026**

**Jenkins Pipeline**

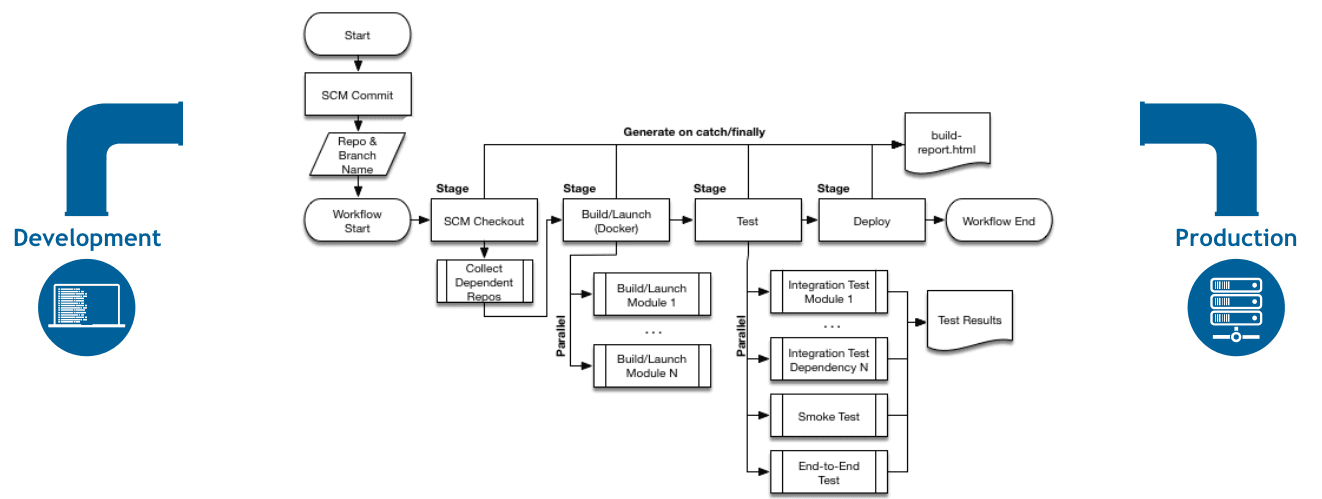
**🔷 Jenkins Pipeline kya hai?**

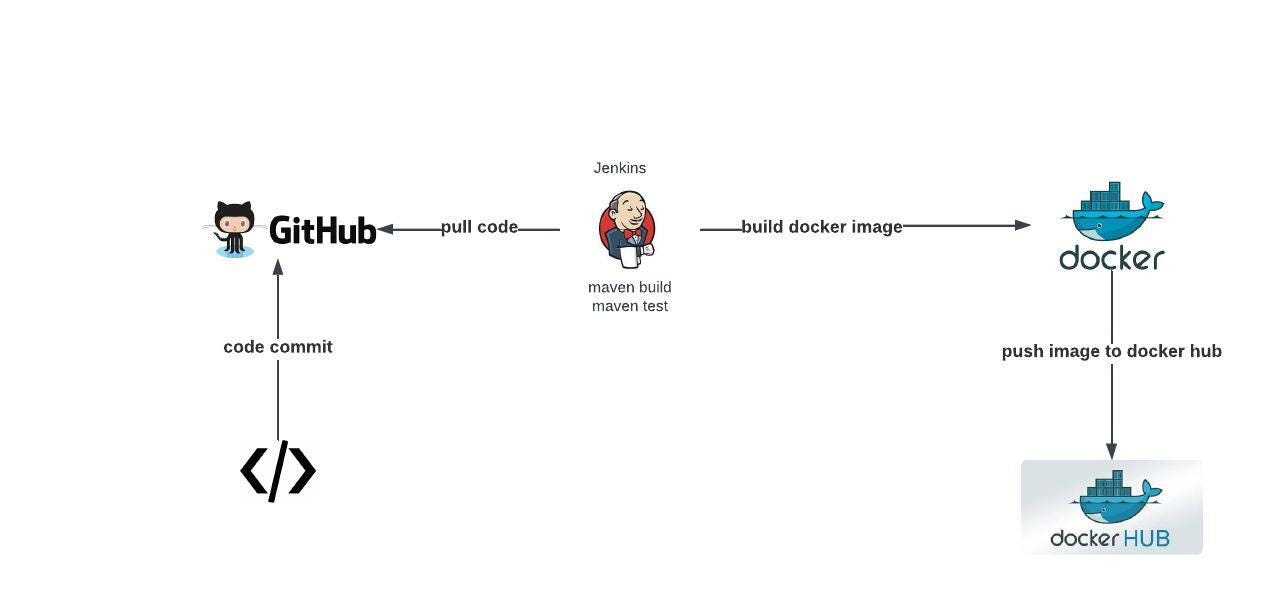
Jenkins Pipeline ka matlab hota hai:

👉 **Build + Test + Deploy ko step-by-step automate karna code ke through**

Matlab:

* Button dabane ki jagah
* **Jenkinsfile likho**
* Jenkins khud sab kaam kare 💪





**Real life example 👇**  
Socho tum maggi bana rahe ho:

1. Paani garam
2. Maggi daalo
3. Masala
4. Serve

Yahi **pipeline** hai – steps fixed order me 💡

**🔷 Jenkinsfile kya hota hai?**

📄 **Jenkinsfile** ek normal text file hoti hai  
📍 Project ke **root folder** me hoti hai  
📍 Isme poora pipeline likha hota hai

👉 Jenkins is file ko padh ke kaam karta hai

**🟢 ✅ Declarative Pipeline kya hoti hai?**

**👉 Declarative Pipeline Jenkins ka ek simple aur structured way hai pipeline likhne ka.  
Isme hum rules ke according pipeline likhte hain.**

**🧠 Simple words:**

**Declarative pipeline ek fixed format hota hai jisme Jenkins ko bataya jata hai kya, kab aur kaise run karna hai.**

**🔹 Declarative ka matlab kya?**

**Declarative = batana (declare karna)**

**Matlab:**

* **Jenkins ko clearly batate ho**
* **Kaunsa agent**
* **Kaunsa stage**
* **Kaunsa step**

**🧱 Declarative Pipeline ka basic structure**

**pipeline {**

**agent any**

**stages {**

**stage('Build') {**

**steps {**

**echo 'Building app'**

**}**

**}**

**}**

**}**

**👉 Ye minimum valid declarative pipeline hai.**

**🔍 Line-by-line explanation (EXAM STYLE):**

| **Line** | **Matlab** |
| --- | --- |
| **pipeline {}** | **Declarative pipeline start** |
| **agent any** | **Jenkins kisi bhi node pe run kare** |
| **stages {}** | **Sare stages yahan likhte hain** |
| **stage('Build')** | **Build naam ka stage** |
| **steps {}** | **Is stage ke commands** |
| **echo** | **Console me message print** |

**✅ Declarative Pipeline ki khas baatein (ADVANTAGES)**

**✔️ Easy to read  
✔️ Fixed structure  
✔️ Beginner-friendly  
✔️ Error kam hote hain  
✔️ Jenkins khud validation karta hai**

**❌ Declarative Pipeline ki limits**

**❌ Complex logic mushkil  
❌ Kam flexible  
❌ Advanced conditions ke liye Scripted pipeline better**

**✅ Scripted Pipeline kya hoti hai?**

**👉 Scripted Pipeline Jenkins ki advanced pipeline hoti hai  
jo Groovy programming language par based hoti hai.**

**🧠 Simple words:**

**Scripted pipeline me hum Jenkins ko program ki tarah instructions dete hain — loops, conditions, variables sab use kar sakte hain.**

**🔹 Scripted ka matlab kya?**

**Scripted = code likhna (programming style)**

**Matlab:**

* **Fixed structure nahi hota**
* **Jo logic chaho likh sakte ho**
* **Jenkins Groovy code execute karta hai**

**🧱 Scripted Pipeline ka basic structure**

**node {**

**stage('Build') {**

**echo 'Building app'**

**}**

**}**

**👉 Ye minimum valid scripted pipeline hai.**

**🔍 Line-by-line explanation**

| **Line** | **Matlab** |
| --- | --- |
| **node {}** | **Jenkins ka agent select** |
| **stage('Build')** | **Build stage** |
| **echo** | **Console me print** |

**🔄 Declarative vs Scripted (CORE DIFFERENCE)**

| **Declarative** | **Scripted** |
| --- | --- |
| **Rule-based** | **Code-based** |
| **Fixed syntax** | **Free-style** |
| **Easy for beginners** | **Needs Groovy knowledge** |
| **Less flexible** | **Highly flexible** |
| **Validation built-in** | **Errors runtime par aate** |

**🧪 Scripted Pipeline ka real example**

**node {**

**stage('Build') {**

**sh 'mvn clean package'**

**}**

**if (env.BRANCH\_NAME == 'main') {**

**stage('Deploy') {**

**echo 'Deploying to production'**

**}**

**}**

**}**

**👉 Ye kaam karta hai:**

* **Build karta hai**
* **Condition check karta hai**
* **Sirf main branch par deploy karta hai**

**❌ Scripted Pipeline ke drawbacks**

**❌ Read karna mushkil  
❌ Errors runtime par  
❌ Maintenance tough  
❌ Freshers ke liye confusing**

👉 **Tum Declarative hi use karo** ✔️

**🔷 Jenkinsfile ko line-by-line samjho 🧠**

pipeline {

👉 Pipeline start ho rahi hai

agent any

👉 Jenkins kisi bhi available machine par run kar sakta hai

stages {

👉 Saare steps yahan likhenge

stage('Build') {

👉 Stage ka naam = Build

steps {

👉 Stage ke andar kya kaam hoga

echo 'Building app'

👉 Console me message print hoga

📤 **Expected Output**

Building app

}

👉 Block close

Now we make demo Jenkins pipeline

**🛠 PRACTICE (REAL PRACTICAL)**

**🔹 STEP 1: Jenkinsfile banao (VS Code / Terminal)**

touch Jenkinsfile

📤 Output:

(no output)

🧠 Matlab:  
👉 Jenkinsfile ban gayi

**🔹 STEP 2: Jenkinsfile open karo**

nano Jenkinsfile

📤 Output:

(blank editor)

🧠 Matlab:  
👉 Ab pipeline code likh sakte ho

**🔹 STEP 3: Pipeline code likho**

pipeline {

agent any

stages {

stage('Build') {

steps {

echo 'Building app'

}

}

}

}

**🔍 Line-by-Line Explanation**

pipeline {

👉 Jenkins pipeline start

agent any

👉 Jenkins kisi bhi available machine par job chalayega

stages {

👉 Saare steps yahan aayenge

stage('Build') {

👉 Stage ka naam **Build**

steps {

👉 Build stage me kaam likhenge

echo 'Building app'

👉 Console me message print karega

Now go on Jenkins dashboard

**🟢 STEP 1: Jenkins open karo**

Browser me likho:

http://localhost:8080

(ya EC2 ho to public-IP:8080)

📤 Output:  
👉 Jenkins Dashboard open

🧠 Matlab:  
👉 Jenkins ready hai job run karne ke liye

**🟢 STEP 2: New Pipeline Job banao**

1️⃣ **New Item** par click  
2️⃣ Job ka naam likho (example):

pipeline-demo

3️⃣ **Pipeline** select karo  
4️⃣ **OK** par click

📤 Output:  
👉 Job configuration page open

**🟢 STEP 3: Jenkinsfile ka code job me daalo**

**Neeche scroll karo → Pipeline section**

* **Definition**:

Pipeline script

* **Script box** me ye paste karo 👇

pipeline {

agent any

stages {

stage('Build') {

steps {

echo 'Building app'

}

}

}

}

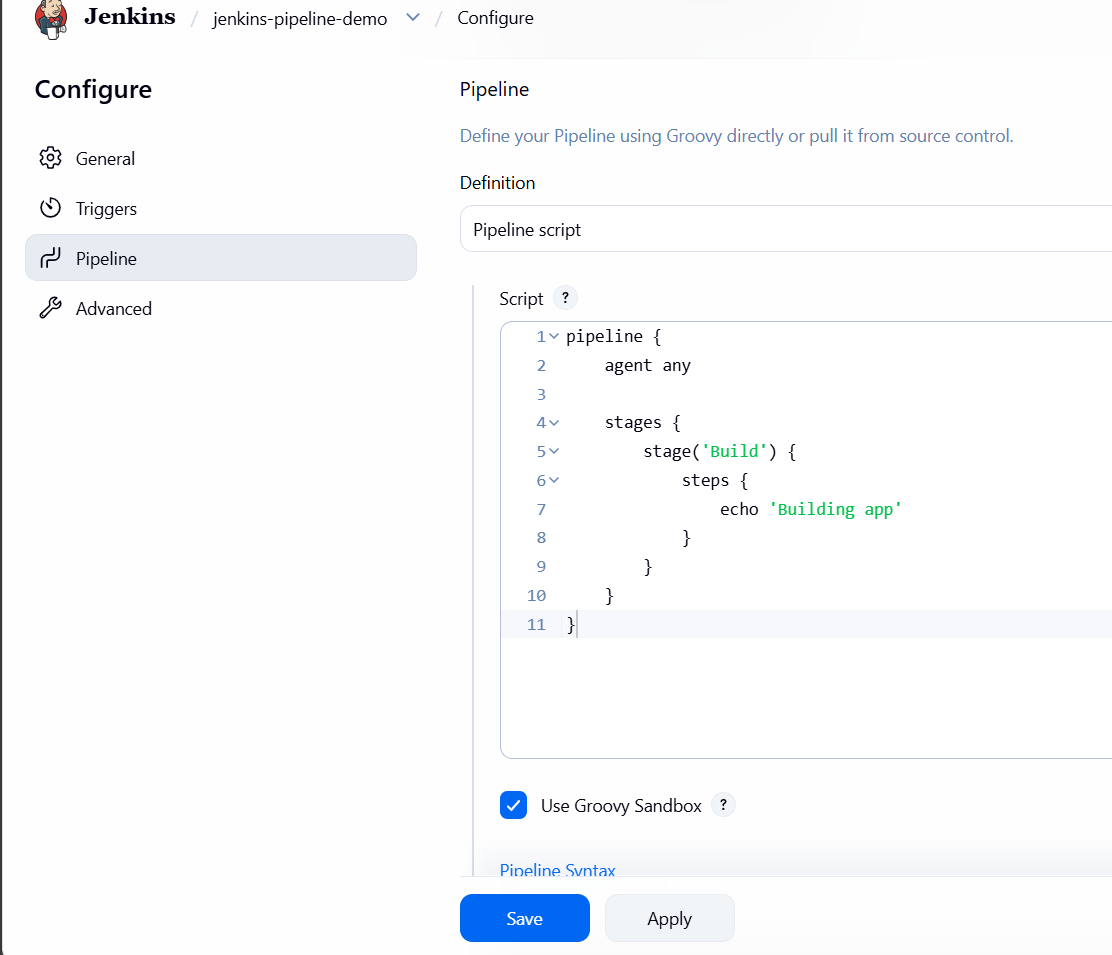
👇 Phir:

* **Save** par click

🧠 Matlab:  
👉 Tumne Jenkins ko bata diya **kya kaam karna hai**

**🟢 STEP 4: Job RUN karo (MOST IMPORTANT)**

1️⃣ Left side me **Build Now** par click karo





Goal is achived hmay Jenkins likhna aagya.

Day-8

Jenkins advanced pipeline

**🔷 Aaj kya seekhenge (CLEAR LIST)**

✔ Multiple stages (Build / Test / Deploy)  
✔ Environment variables  
✔ Parameters (user input)  
✔ Post actions (success / failure)  
✔ **Real-world CI pipeline**

Multiple stages:

👉 Multiple stages ka matlab hai pipeline ko chhote-chhote logical steps me tod dena.  
Har stage ek specific kaam karta hai.

🧠 Simple words:

Jenkins Pipeline me multiple stages isliye hoti hain taaki build process **step-by-step, clear aur manageable** ho.

**🔹 Stage ka matlab kya?**

**Stage = ek phase / ek step**

Jaise: build ,test &deploy karna

**🧱 Multiple Stages ka basic example**

pipeline {

agent any

stages {

stage('Build') {

steps {

echo 'Build stage running'

}

}

stage('Test') {

steps {

echo 'Test stage running'

}

}

stage('Deploy') {

steps {

echo 'Deploy stage running'

}

}

}

}

**📜 Jenkins Declarative Pipeline Script**

pipeline {

👉 **Pipeline ka start**  
Jenkins ko batata hai: “yahan se pipeline define ho rahi hai”.

agent any

👉 **Agent select karna**  
Jenkins pipeline **kisi bhi available node/agent** par run ho sakti hai.

stages {

👉 **Stages block start**  
Pipeline ke saare steps (Build, Test, Deploy) yahin likhe jaate hain.

stage('Build') {

👉 **Build naam ka stage**  
Is stage me build related kaam hoga (compile, package, etc.).

steps {

👉 **Steps block**  
Is stage ke andar **actual commands** yahin likhi jaati hain.

echo 'Build stage running'

👉 **Console message print karega**  
Jenkins console output me ye text dikhayega.

}

👉 **Steps block end**  
Build stage ke steps yahin khatam.

}

👉 **Build stage end**

stage('Test') {

👉 **Test naam ka stage**  
Is stage me testing related kaam hota hai.

steps {

👉 **Test stage ke steps start**

echo 'Test stage running'

👉 **Console me Test stage ka message print**

}

👉 **Test stage ke steps end**

}

👉 **Test stage end**

stage('Deploy') {

👉 **Deploy naam ka stage**  
Is stage me application deploy ki jaati hai.

steps {

👉 **Deploy stage ke steps start**

echo 'Deploy stage running'

👉 **Console me Deploy stage ka message print**

}

👉 **Deploy stage ke steps end**

}

👉 **Deploy stage end**

}

👉 **Stages block end**  
Saare stages yahin complete hote hain.

}

👉 **Pipeline ka end**  
Jenkins ko batata hai pipeline yahin finish hoti hai.

👉 Ye pipeline **3 stages** me run hoti hai:  
1️⃣ Build  
2️⃣ Test  
3️⃣ Deploy

**🔍 Line-by-line simple explanation**

| **Stage** | **Kya kaam hota hai** |
| --- | --- |
| Build | Code compile / package |
| Test | Code test / verify |
| Deploy | App server par deploy |

**🔄 Jenkins multiple stages ko kaise chalata hai?**

➡️ **Order me (top to bottom)**

1. Build complete
2. Build SUCCESS → Test start
3. Test SUCCESS → Deploy start
4. Koi stage FAIL → aage ke stages **run nahi hote**

🧠 Rule:

Jenkins next stage tabhi chalata hai jab previous stage SUCCESS ho.

**🧪 Real-life analogy (yaad rakhne ke liye)**

Socho tum **khana bana rahe ho** 🍳

1️⃣ Sabzi kaatna (Build)  
2️⃣ Pakana (Test)  
3️⃣ Serve karna (Deploy)

Agar sabzi hi na kati ❌  
to pakana aur serve **possible nahi** 😄

**✅ Multiple stages kyun zaroori hoti hain? (ADVANTAGES)**

✔️ Pipeline clear dikhti hai  
✔️ Error easily pakad me aata hai  
✔️ Resume / interview friendly  
✔️ Visualization achha hota hai (Blue Ocean)  
✔️ Real CI/CD flow follow hota hai

**❌ Agar multiple stages na ho?**

Sab kuch ek hi stage me likh doge:

* Confusing ho jayega
* Debug mushkil
* Industry practice follow nahi hogi

New item banyoh; Jenkins-multiple-stage-1

Pipeline select karoh then ok

Pipeline p click : usmay pipelinscript p yeh copy kardoh

pipeline {

agent any

stages {

stage('Build') {

steps {

echo 'Build stage running'

}

}

stage('Test') {

steps {

echo 'Test stage running'

}

}

stage('Deploy') {

steps {

echo 'Deploy stage running'

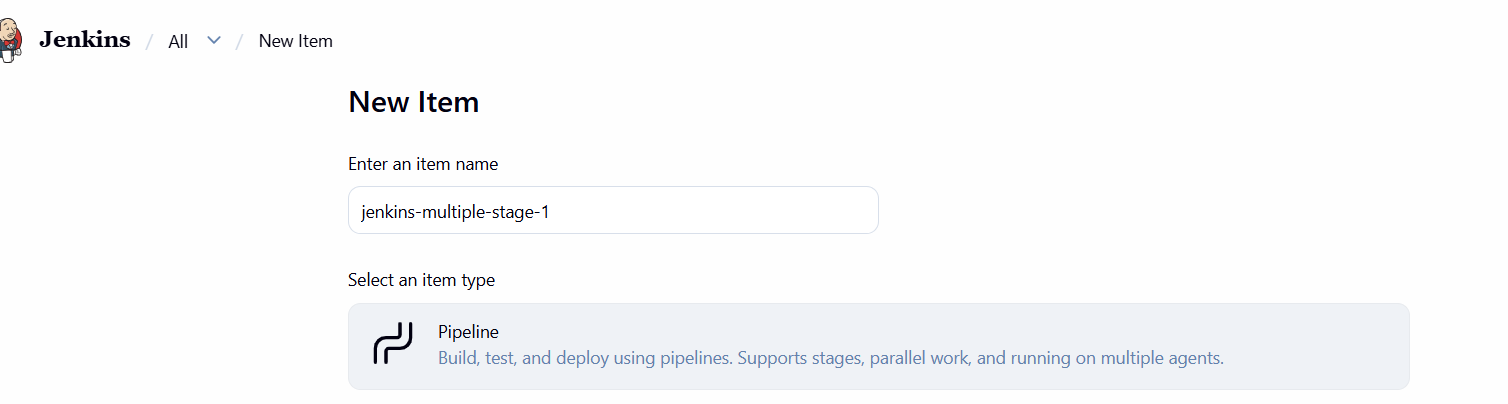
}

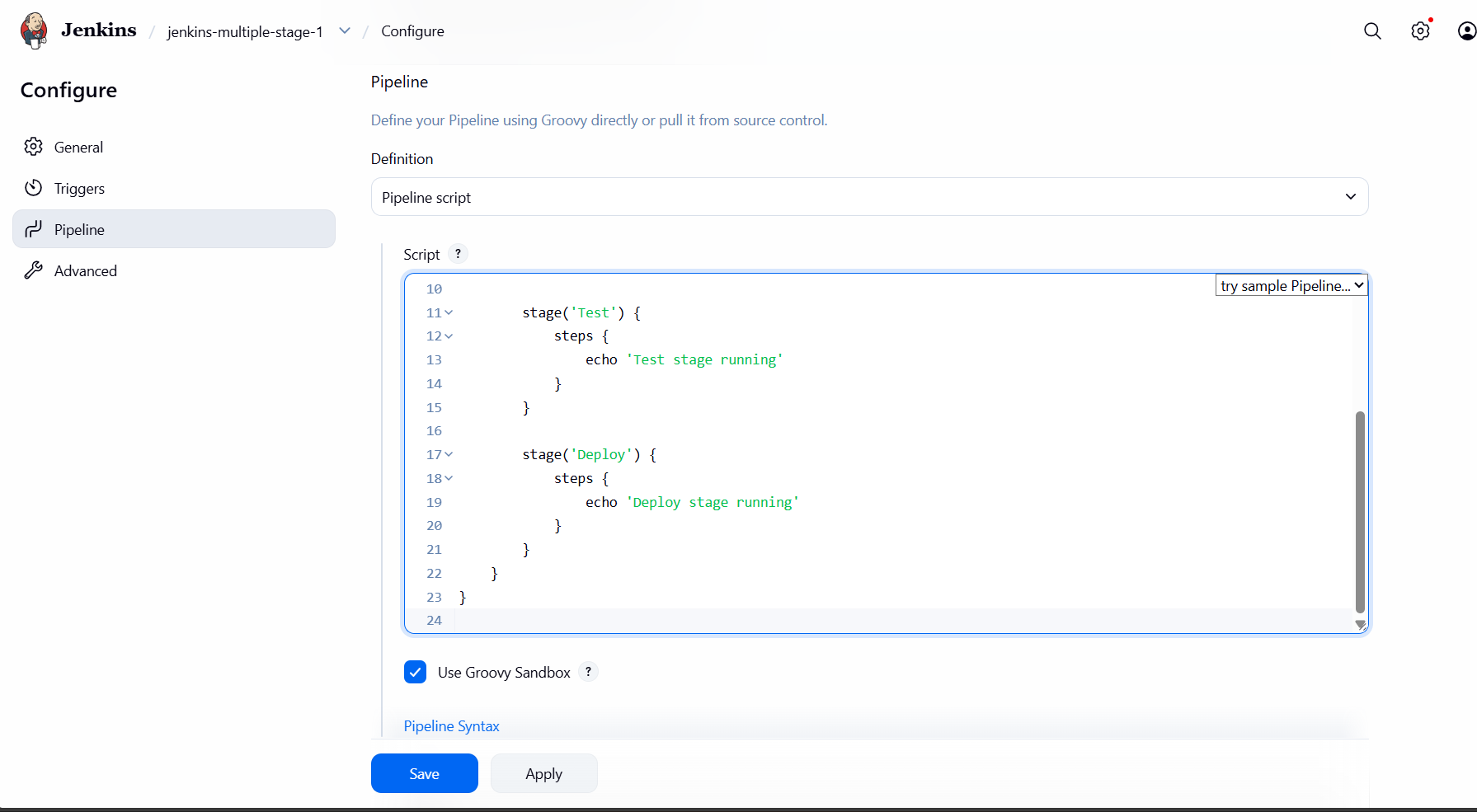
}

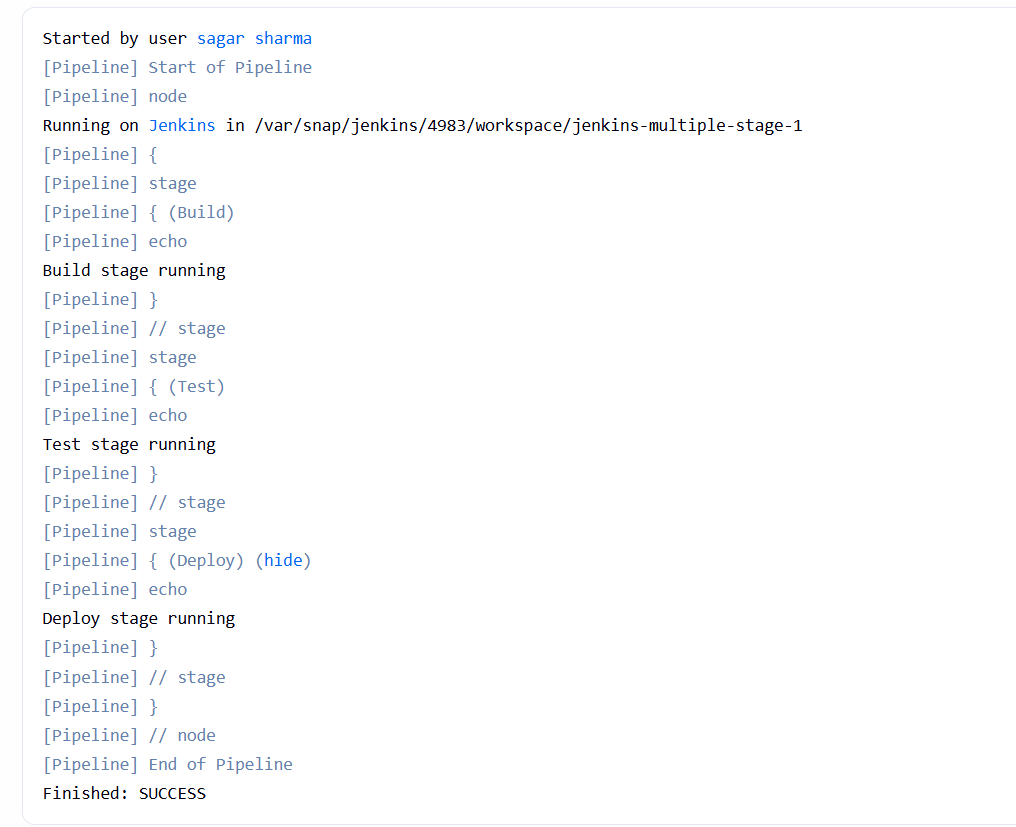
}

}

Then save build now







(2.) **Environment Variables kya hote hain?**

**Environment Variables** wo **key–value pair** hote hain  
jo **system / Jenkins / application** ko runtime par information dete hain.

👉 Ye values **code ke bahar** hoti hain  
👉 Har jagah same cheez likhne se bachati hain  
👉 Configuration easy banati hain

**🧠 Simple example (real life)**

Socho:

* Tumhara **naam** = Sagar
* Tumhara **ghar** = Panipat

Isko system aise yaad rakhta hai:

NAME=Sagar

CITY=Panipat

Yehi concept **environment variables** ka hai.

**💻 Linux / Ubuntu example**

echo $HOME

**Output:**

/home/ubuntu

👉 $HOME ek environment variable hai  
👉 Jo user ka home directory batata hai

**🧪 Kuch common Environment Variables**

| **Variable** | **Matlab** |
| --- | --- |
| HOME | User ka home folder |
| PATH | Commands ka path |
| USER | Current user |
| JAVA\_HOME | Java ka install path |
| WORKSPACE | Jenkins job ka folder |

**>>>>>>>>>>>>>>>**

**>>>>>>>>>>>.**

**>>>>>>>>>>>>>>>**

**🧩 Jenkins me Environment Variables:**

Jenkins **automatically** kuch variables deta hai 👇

echo env.WORKSPACE

👉 Current job ka workspace path

Echo env.BUILD\_NUMBER

👉 Build ka number

echo env.JOB\_NAME

👉 Jenkins job ka naam

**🧪 Jenkins pipeline example for environment variable:**

pipeline {

agent any

stages {

stage('Env Demo') {

steps {

echo "Job name is ${env.JOB\_NAME}"

echo "Build number is ${env.BUILD\_NUMBER}"

echo "Workspace is ${env.WORKSPACE}"

}

}

}

}

👉 Ye Jenkins console me environment values print karega.

**🛠 Custom Environment Variable (khud banana)**

pipeline {

agent any

environment {

APP\_NAME = "MyApp"

VERSION = "1.0"

}

stages {

stage('Print') {

steps {

echo "App = ${env.APP\_NAME}"

echo "Version = ${env.VERSION}"

}

}

}

}

**🎯 Environment Variables kyu use karte hain?**

✔ Code clean rehta hai  
✔ Same code dev/test/prod me use hota hai   
✔ Secrets (password, token) hide kiye ja sakte hain  
✔ Automation easy ho jata hai

Now we make Jenkins environment variable

Create item: Jenkins-demo-env-variable

Pipeline selct karoh then ok

Pipeline p click karoh and pipeline-script valay box m yeh code paste kardoh :

pipeline {

agent any

stages {

stage('Env Demo') {

steps {

echo "Job name is ${env.JOB\_NAME}"

echo "Build number is ${env.BUILD\_NUMBER}"

echo "Workspace is ${env.WORKSPACE}"

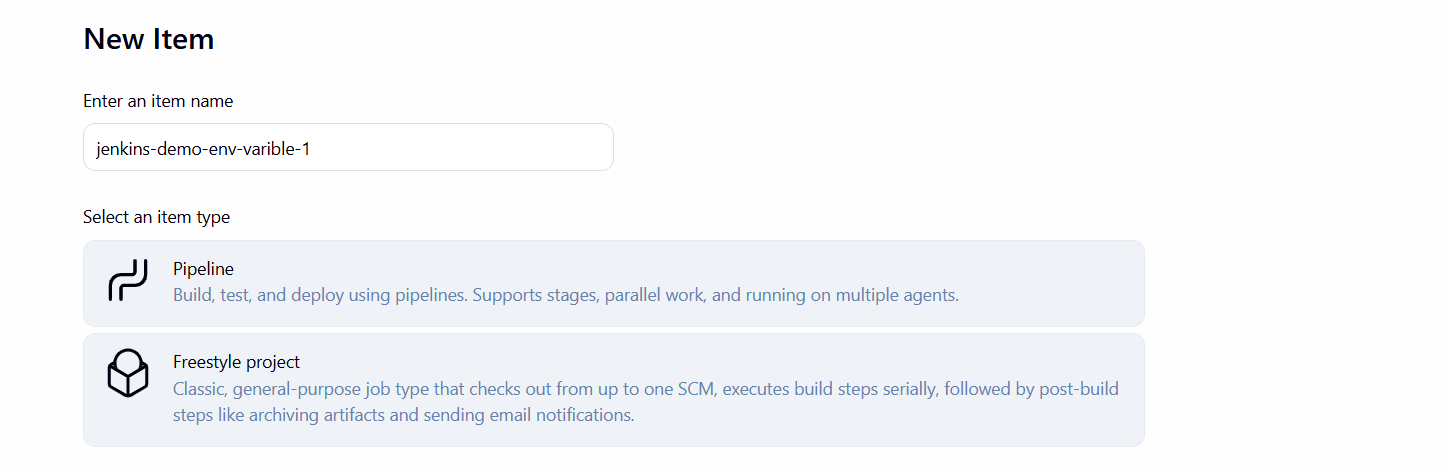
}

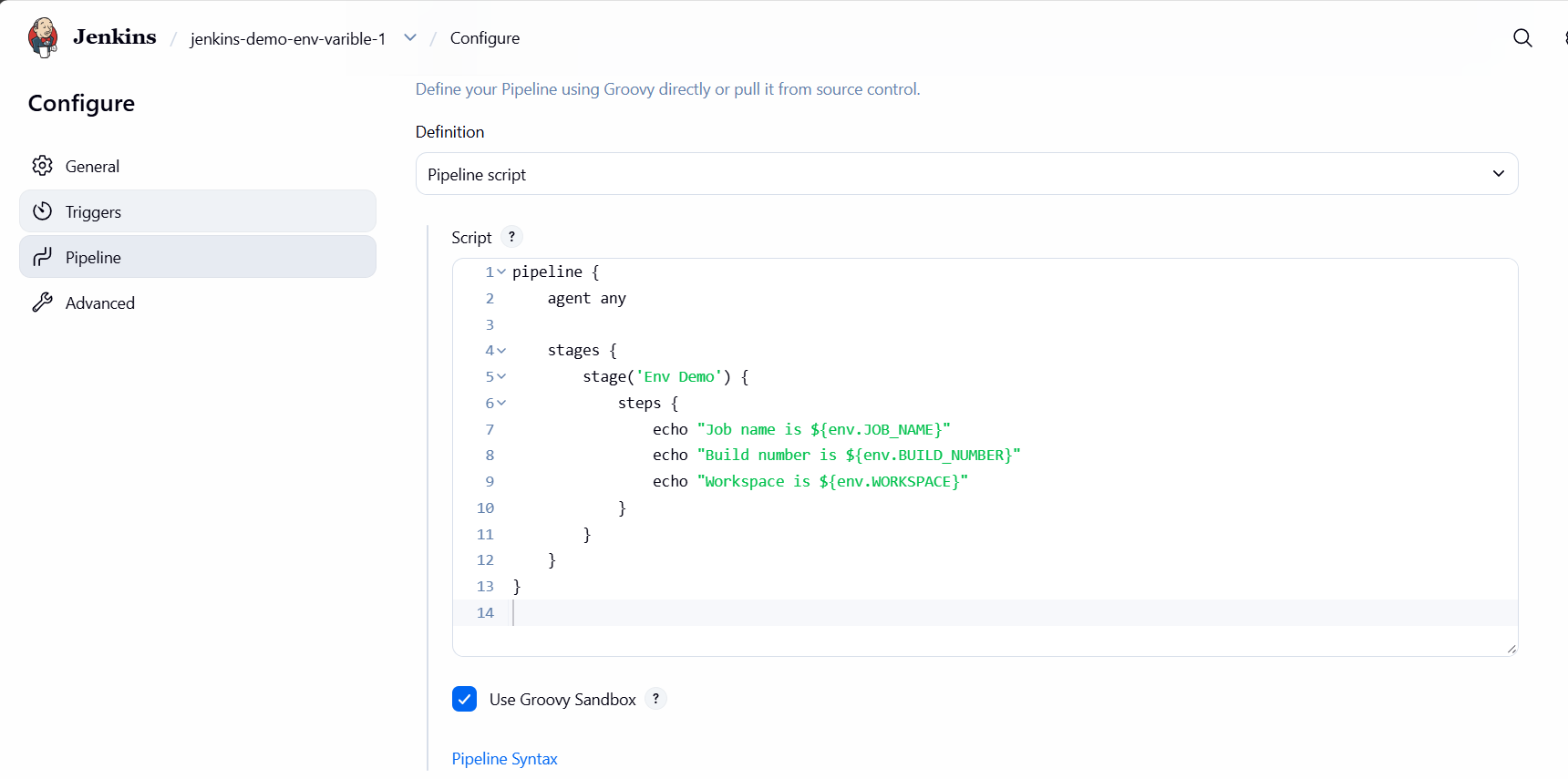
}

}

}

then save then bulid now







>>now we make customize env-varible sab sum bus

Item: Jenkins-customize-env-variable-demo then sab same

pipelinscript valay box m yeh code paste kardoh:

pipeline {

agent any

environment {

APP\_NAME = "MyApp"

VERSION = "1.0"

}

stages {

stage('Print') {

steps {

echo "App = ${env.APP\_NAME}"

echo "Version = ${env.VERSION}"

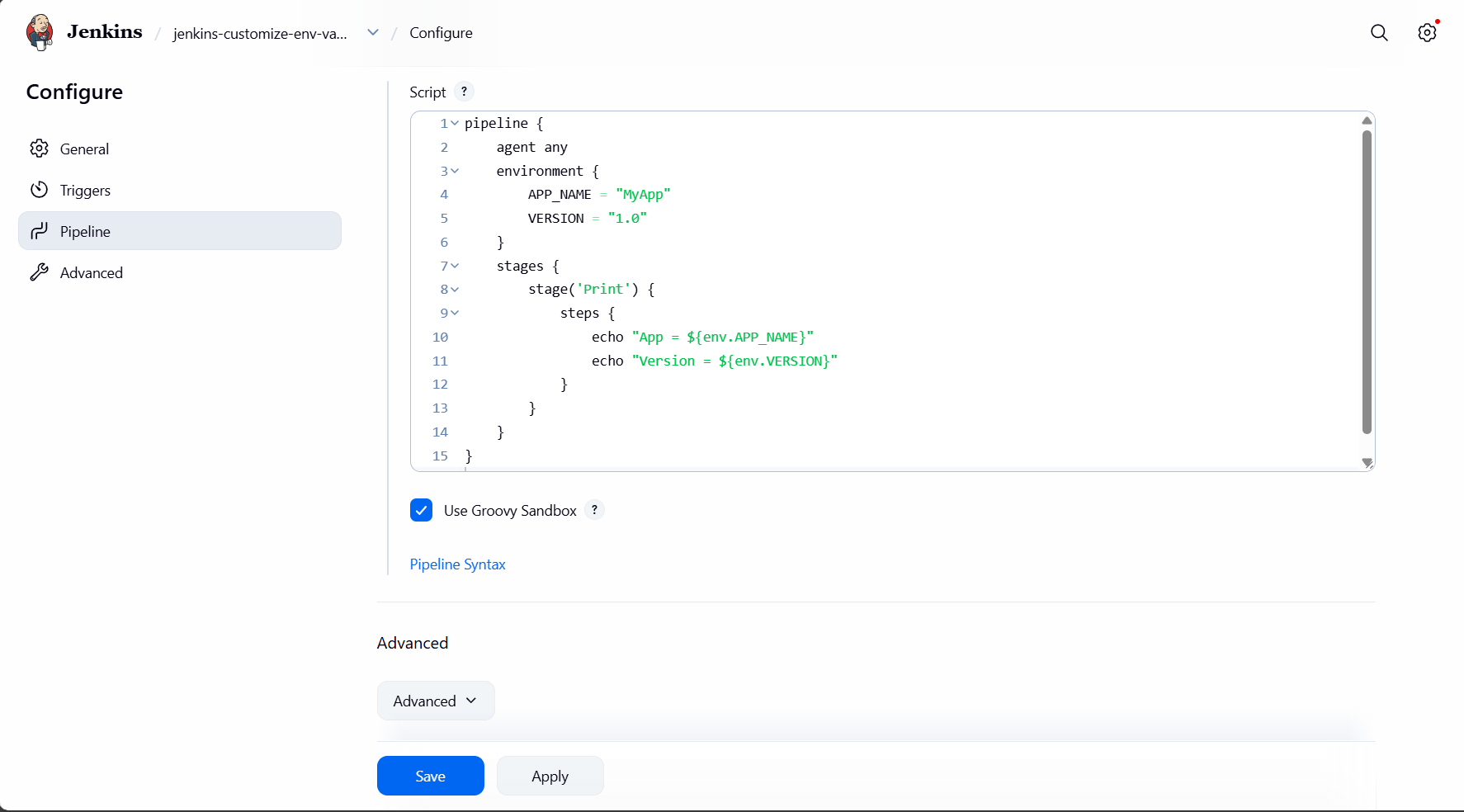
}

}

}

}

Then save then bulid now





3.)parameters(user input)ky hotay hai:

**Parameters** wo inputs hote hain  
jo **user build time par deta hai**,  
aur Jenkins pipeline un values ko **use karta hai**.

👉 Har build me alag value ho sakti hai  
👉 Same pipeline ko flexible banata hai

Iskay 3 types hotay hai

(i)string (ii) choice (iii) boolean parameter

**🧠 Real-life example**

Socho:

* Tumhe poocha jaaye:

“Deploy kis environment me karna hai?”

Options:

* dev
* test
* prod

👉 Ye hi **parameter** hai.

**🛠 Jenkins Pipeline me Parameters ka syntax**

parameters {

...

} yeh block pipeline ke start m likha jata hai

**✅(i) : String Parameter ky hota hai:**

String Parameter wo parameter hota hai  
jisme user text (string) input karta hai  
jab Jenkins build start hota hai.

👉 User koi bhi text value de sakta hai  
👉 Jaise: app name, version, branch name, server name

🧠 Real-life example

Socho Jenkins tumse poochta hai:

“Application ka naam kya hai?”

User likhta hai:

my-app

👉 Ye hi string parameter hai.

🛠 Jenkins Pipeline me String Parameter

✅ Basic Syntax

string(name: 'APP\_NAME', defaultValue: 'MyApp', description: 'Application name')

complete working script example: yahi

pipeline {

agent any

parameters {

string(name: 'APP\_NAME', defaultValue: 'MyApp', description: 'Application name')

}

stages {

stage('Print') {

steps {

echo "App name is ${params.APP\_NAME}"

}

}

}

}

**Build time par:**

* Jenkins tumse poochega **APP\_NAME**
* Jo value doge wahi print hogi

**Vahi same hai item: Jenkins-string-prarmeter then pipeline then save then pipe line script valay box may yeh code paste kardoh then build now then output**

****

**✅ (ii): Choice Parameter (MOST USED 🔥):**

**Choice Parameter wo parameter hota hai  
jisme user ko fixed options (dropdown) diye jaate hain,  
aur user sirf unhi me se ek choose karta hai.**

**👉 User kuch galat type nahi kar sakta  
👉 Production jobs me sabse zyada use hota hai**

**🧠 Real-life example**

**Jenkins poochta hai:**

**Deploy kis environment me karna hai?**

**Dropdown:**

* **dev**
* **test**
* **prod**

**👉 Ye hi choice parameter hai.**

**🛠 Jenkins Pipeline me Choice Parameter**

**✅ Syntax**

**choice(name: 'ENV', choices: ['dev', 'test', 'prod'], description: 'Select environment')**

**✅ Complete Working Example:**

pipeline {

agent any

parameters {

choice(name: 'ENV', choices: ['dev', 'test', 'prod'], description: 'Select environment')

}

stages {

stage('Deploy') {

steps {

echo "Deploying to ${params.ENV} environment"

}

}

}

}

👉 Build start karte hi dropdown aayega  
👉 User environment choose Karega

**✅ (iii): Boolean Parameter (YES / NO):**

Boolean Parameter wo parameter hota hai  
jisme sirf do values hoti hain:

* ✔️ true (YES)
* ❌ false (NO)

👉 Jenkins me ye checkbox ke form me dikhta hai.

🧠 Real-life example

Jenkins poochta hai:

Tests run karne hain? ✔️

* Checkbox tick → true
* Checkbox untick → false

👉 Ye hi boolean parameter hai.

🛠 Jenkins Pipeline me Boolean Parameter

✅ Syntax

booleanParam(

name: 'RUN\_TESTS',

defaultValue: true,

description: 'Run tests or not

**✅ Complete Working Example**

**Phele item: jenkins-boolean-parameter-demo then**

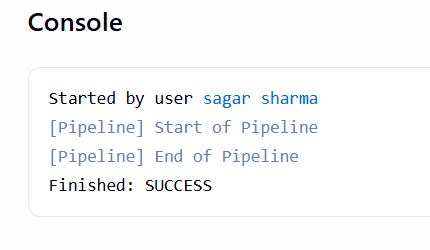
**Pipeline then ok then pipeline select karkay pipeline-script valay box m yeh code paste karkay save then build now**

parameters {

booleanParam(name: 'RUN\_TESTS', defaultValue: true, description: 'Run tests or not')

}

Output checkbox milega:



**🧪 Parameters vs Environment Variables:**

| **Parameter** | **Environment Variable** |
| --- | --- |
| User input leta hai | System/Jenkins deta hai |
| Build start par set hota | Build ke dauraan available |
| ${params.X} | ${env.X} |

(4.) post actions (success/failure) ky hota hai?

Post actions wo steps hote hain  
jo pipeline ke complete hone ke baad run hote hain,  
chahe build SUCCESS ho ya FAILURE.

👉 Ye cleanup, notification, logging jaise kaam ke liye hote hain.

🧠 Simple words me:

“Build ke baad kya karna hai” = post actions

🧱 Post block ka syntax (Declarative Pipeline):

post {

success {

// jab build SUCCESS ho

}

failure {

// jab build FAIL ho

}

always {

// har case me

}

} 👉 post {} block stages ke baad likha jaata hai.

Now make new item: Jenkins-post-actions-pipeline-demo then pipeline p click ok then pipeline p click karoh then pipeline script box m yeh code paste kardoh then paste them save then buils now

Basic Working Example code:>>>>>>> nichay hai:pura nichay tak hai: avoid brackets

[

pipeline {

agent any

stages {

stage('Build') {

steps {

echo 'Building application'

}

}

}

post {

success {

echo 'Build SUCCESS hua'

}

failure {

echo 'Build FAIL hua'

}

always {

echo 'Pipeline finished'

}

}

}

]avoid brackets



🔍 Har post condition ka matlab

| Post Action | Kab run hota hai |
| --- | --- |
| success | >>> Jab build SUCCESS ho |
| failure | >.>> Jab build FAIL ho |
| always | >>> Har haal me |
| unstable | >>> Tests fail hue |
| aborted | >>> Build manually stop hua |
| changed | >>> Previous build se result change |

(5) real-world ci pipeline kya hai ?

Continuous Integration (CI) ka matlab hai:

👉 Jaise hi developer code change kare,  
👉 automatically build + test chal jaye,  
👉 taaki error jaldi pakad me aa jaye.

🧠 Simple words:

Real-world CI me code manually build nahi hota, sab kuch automated hota hai.

🧑‍💻 Company me REAL CI ka flow (STEP-BY-STEP)

🔁 Real life pipeline:

Developer → GitHub → Jenkins → Build/Test → Result

1️⃣ Developer code change karta hai

* New feature
* Bug fix
* Small update

git commit -m "login bug fixed"

git push origin main

2️⃣ Code GitHub par push hota hai

* GitHub repo updated
* Webhook trigger hota hai

👉 Jenkins ko signal milta hai:

“Naya code aaya hai”

3️⃣ Jenkins automatically start hota hai

Jenkins:

* Repo clone karta hai
* Jenkinsfile read karta hai

4️⃣ CI Pipeline run kaisay hoti hai

Typical real-world CI stages:

| Stage | Kaam |
| --- | --- |
| Checkout | GitHub se code lena |
| Build | mvn clean package |
| Test | Unit tests |
| Quality | SonarQube check |
| Package | JAR/WAR banana |

5️⃣ Result milta hai (FAST feedback)

✅ Success:

* Code OK
* Team confident

❌ Failure:

* Build fail
* Jenkins turant batata
* Developer fix karta hai

Jab hum Jenkins per Jayda kam kartay hai toh jenkins per overload ajaata hau due to which jenkins m gol gol ghumnay lag jaata hai pipeline ka bulid now k baad that means Jenkins k pass space khatam ho gayi usay aur space chiy then we install Jenkins using war method.

🚀 FINAL FIX: Jenkins via WAR (Guaranteed)

1️⃣ Java already hai (check) java -version Agar nahi: sudo apt install openjdk-17-jdk -y

2️⃣ Jenkins WAR download karo wget <https://get.jenkins.io/war-stable/latest/jenkins.war>

3️⃣ Jenkins start karo java -jar jenkins.war --httpPort=8080 Terminal open rehne do.

4️⃣ Browser me open http://<Elastic-IP>:8080 Unlock password milega terminal me.

And maven install kardoh using

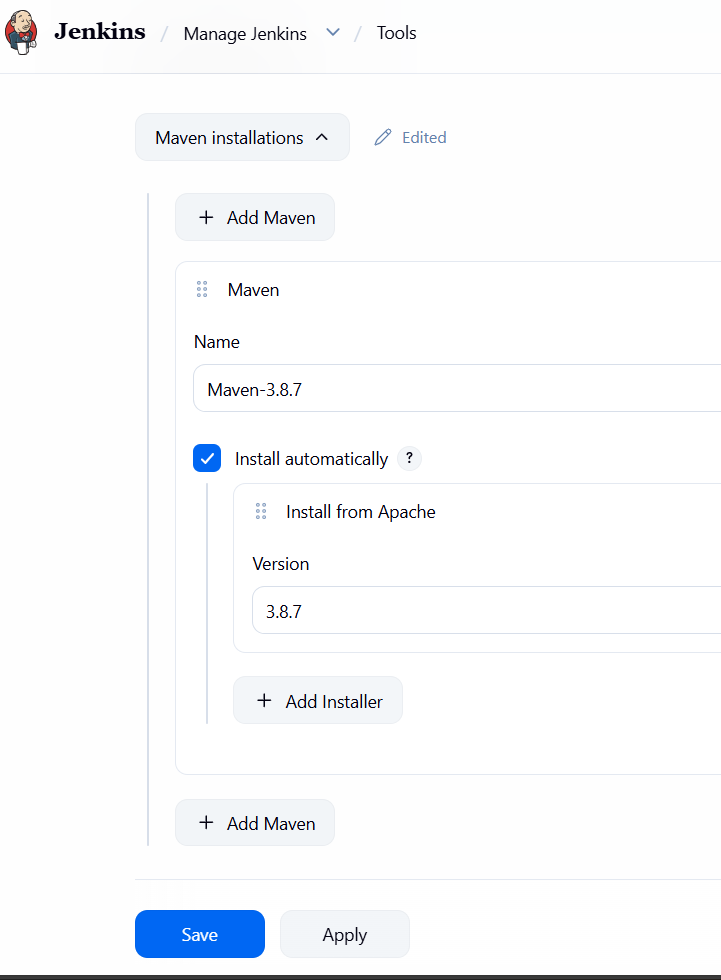
sudo apt install maven -y

check: mvn -version

yeh real world pipeline chlanay say phele manage Jenkins m jayoh then click tools then scroll down vha maven installation per click karoh vha type karoh

Maven-3.8.7 then version select karoh 3.8.7 then save

Like this



Now we make

new item: Jenkins-real-world-maven-pipeline-demo then pipeline p click karkau then ok then pipeline p click karoh then pipeline script box m yeh nichy vala code paste kardoh

🧪 Real-world Jenkins CI Pipeline example:

[

pipeline {

agent any

tools {

maven 'Maven-3.8.7'

}

stages {

stage('Checkout') {

steps {

git branch: 'main',

url: 'https://github.com/sagarvashist19-boop/jenkins-git-demo-2.git'

}

}

stage('Build') {

steps {

sh 'pwd'

sh 'ls -l'

sh 'mvn clean package -DskipTests'

}

}

stage('Test') {

steps {

sh 'mvn test'

}

}

}

post {

success {

echo 'CI SUCCESS'

}

failure {

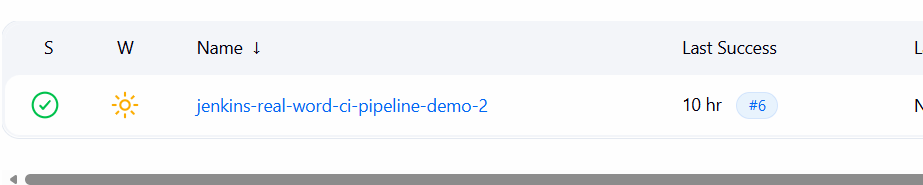
echo 'CI FAILED – Fix immediately'

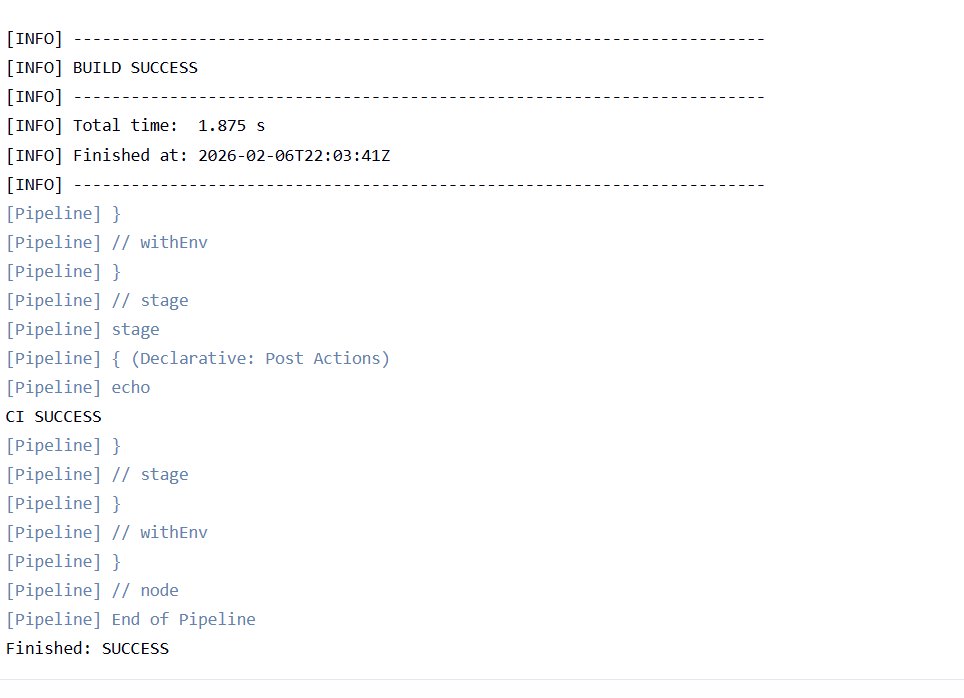
}

}

}

]avoid brackets





**Code ki har line ka matlab:**

**pipeline {**

👉 Jenkins pipeline ka **start**.  
Iske andar poora CI process define hota hai.

**🔵 agent any**

👉 Jenkins ko bolo:  
**“Kisi bhi available node / agent pe job chala do”**  
(master ya koi slave – jo free ho)

**🔵 tools {** 👉 Jenkins ke tools define karte hain jo build ke time use honge.

**🔵 maven 'Maven-3.8.7'**

👉 Jenkins me jo Maven **global tool configuration** me add hai,  
uska **name = Maven-3.8.7** use hoga.

📌 Matlab:  
mvn command automatically available ho jaayegi.

**🔵 stages {**

👉 Pipeline ko **steps (stages)** me divide karta hai  
(Checkout → Build → Test)

**🟡 stage('Checkout')**

👉 GitHub se **source code download** karne ka stage.

**🔹 git branch: 'main',**

👉 Repository ki **main branch** checkout karega.

**🔹 url: 'https://github.com/...git'**

👉 GitHub repository ka URL jahan se code aayega.

📌 Result:  
Project Jenkins workspace me aa jaata hai.

**🟡 stage('Build')**

👉 Project ko **compile + package** karne ka stage.

**🔹 sh 'pwd' >**👉 Current working directory print karta hai.  
(Matlab Jenkins ka **workspace path**)

**🔹 sh 'ls -l'**

👉 Workspace ke **sab files/folders list** karta hai.  
(Confirm karne ke liye ki code aaya hai)

**🔹 sh 'mvn clean package -DskipTests'**

👉 Maven command:

* clean → purana build delete
* package → JAR / WAR banata hai
* -DskipTests → tests **skip** kar deta hai (fast build)

**🟡 stage('Test')**

👉 Unit testing ke liye stage.

**🔹 sh 'mvn test'**

👉 Maven ke **JUnit / Test cases run** karta hai.

📌 Agar koi test fail → pipeline fail ❌

**🔵 post {**

👉 Pipeline ke **result ke baad** kya karna hai.

**🟢 success {**

👉 Agar sab stages successfully pass ho jaayein.

**🔹 echo 'CI SUCCESS'**

👉 Console me message print karega.

**🔴 failure {**

👉 Agar koi bhi stage fail ho jaaye.

**🔹 echo 'CI FAILED – Fix immediately'**👉 Error message show karega taaki turant issue fix ho.

**🔵 }**

👉 Pipeline ka **end** 🎉

**🧠 Pipeline Flow (Samajhne ke liye)**

GitHub se code

↓

Maven build

↓

Test cases

↓

SUCCESS / FAILURE message

🧠 Real-world CI ka main goal

✔ Bugs jaldi milen  
✔ Integration issues kam ho  
✔ Team ka time bache  
✔ Stable codebase rahe  
✔ Faster delivery

❌ CI ke bina kya hota hai?

* Manual builds
* Late bug detection
* “Works on my machine” problem
* Deployment failures

Day 9 – Jenkins Security & Agents

**🔴 Kya Jenkins dobara install karna padega kal sab kuch close kardiya tha:**

❌ **Nahi**, agar sirf:

* laptop / EC2 **band** kiya tha
* terminal / browser **close** kiya tha

👉 Jenkins **already installed hi rehta hai**.

**✅ Jenkins ko wapas open kaise kare?**

Browser me likho:

http://<server-ip>:8080

Agar local machine hai:

http://localhost:8080

**Jenkins WAR ka matlab**

* Jenkins **service ke form me install nahi**
* Tu sudo systemctl status Jenkins status nahi dikhara tha toh hum war jenkins use kartay hai

> “Status active hai, phir kal WAR Jenkins kyu install kiya tha?

🧠 Short me jawab

👉 Tumne galti se Jenkins ke 2 tareeke mix kar diye

* ek Service (apt install)
* ek WAR (manual run)

**🧩 Kal actually kya hua tha (real story)**

**🟢 Step 1: Tumne Jenkins install kiya (APT se)**

sudo apt install jenkins

✔ Isse kya hua:

* Jenkins **system service** ban gaya
* Auto-start enable ho gaya
* Port **8080** lock ho gaya
* Background me chalne laga

👉 Isi liye aaj:

systemctl status jenkins = active

**🔴 Step 2: Tumhe laga Jenkins start nahi hua**

* Browser try kiya → nahi khula (SG / patience / delay)
* Tumne socha “WAR try karte hain”

👉 Yahin se confusion start hua 😄

**🔴 Step 3: Tumne WAR Jenkins download kiya**

wget jenkins.war

java -jar jenkins.war

❌ Problem:

* Port 8080 already service Jenkins le chuka tha
* WAR Jenkins fail → Jetty error
* Tum kill karte rahe, service auto-restart hoti rahi

Tumne Service Jenkins ko samjha hi nahi, aur upar se WAR Jenkins start karne ki koshish kar di.

**✅ Ab FINAL decision (important)**

**🔥 OPTION A (RECOMMENDED – REAL WORLD)**

👉 **Service Jenkins use karo**

✔ Kya karna hai:

sudo systemctl status jenkins

Browser:

http://<EC2-PUBLIC-IP>:8080

❌ WAR ko bhool jao

**>> Jenkins ki space kyun khatam hoti hai?**

Mostly ye jagah pe space khatam hoti hai:

**📂 Jenkins Workspace**

/var/lib/jenkins/workspace/

**📂 Jenkins Build Logs**

/var/lib/jenkins/jobs/

**📂 Old builds (sabse bada reason)**

Har build:

* logs
* artifacts
* reports  
  save karta rehta hai

**🟢 4️⃣ Space khatam ho jaaye toh kya karein? (IMMEDIATE FIX)**

**✅ Disk space check**

df -h

**✅ Large folders dekho**

du -sh /var/lib/jenkins/\*

**✅ Old builds delete (sabsay BEST METHOD)**

Jenkins UI →  
**Job → Configure → Discard Old Builds**

Set karo:

* Max builds: 5 ya 10
* Max days: 7

📌 Ye **permanent solution** hai ✅

**Jenkins ki space kabhi khatam na ho – BEST PRACTICES :**

**✅ 1. Discard Old Builds (MOST IMPORTANT)**

✔ Har job me enable karo

**✅ 2. Maven clean use karo**

Tum already use kar rahe ho 👌

mvn clean package

👉 Purana target folder delete ho jaata hai

**Today Goal: day 9 jenkins security &agents:**

✅ Jenkins secure  
✅ Credentials protected  
✅ Agents connected  
✅ Master load kam  
✅ Real-world DevOps setup

1️⃣ Jenkins Users & Roles (Security ka base):

🔐 Jenkins Users kya hote hain?

* Jenkins me multiple users ho sakte hain
* Har user ko limited permissions di jaati hain
* Isse unauthorized access rukta hai

🧩 Default situation

* By default Jenkins me sirf Admin hota hai

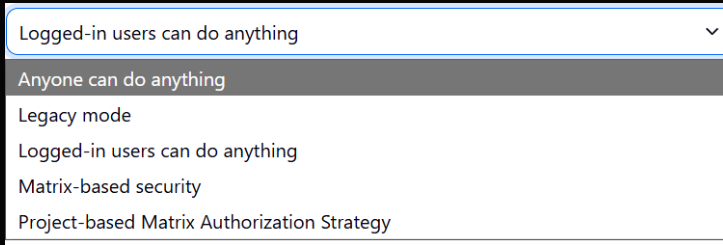
🔹 Enable security (agar pehle se nahi hai)

Dashboard → Manage Jenkins → Security → Security

✔ Check:

* ✅ Enable security
* Security Realm: Jenkins’ own user database
* Authorization: Role-Based Strategy (BEST)

Autjorization m scroll down karoh aur dekho role-based-strategy



Iskay anadar role based strategy nahi hai humay usay install karna hoga .

Ab ky karoh

>>Dashboard → Manage Jenkins → Plugins → Available plugins seach karoh Role-based Authorization Strategy select karoh install without restart

Ab vs code m >>type sudo systemctl restart Jenkins

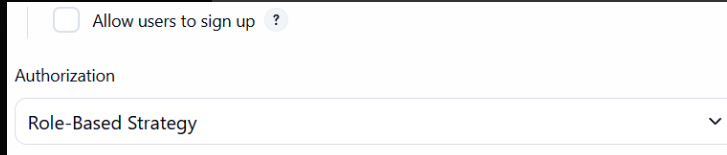
then wait 1-2min

then dubara security page open karoh

>Manage Jenkins → Security → Security

>Authorization dropdown me ab ye dikhna chahiye:

Role-Based Strategy



Role-Based Authorization Strategy Jenkins me users ko specific roles ke basis par permissions assign karne ke liye use hoti hai.

our enable security done.

2️⃣ Jenkins Roles (Admin / Developer / Viewer):

🛠 Plugin install

Manage Jenkins → Plugins → Available

* 🔍 Role-based Authorization Strategy
* Install + Restart

Yeh kaam humnay upper karliya.

🎭 Roles example

| Role | Permissions |
| --- | --- |
| Admin | Full access |
| Developer | Job create, build |
| Viewer | Read-only |

📍 now we do Role setup path :

Manage Jenkins → Security section m dekho → Manage and Assign Roles p click karengay

Nichay dekho ky karna hai ab.

Agar kabhi manage and assign role n dikhay toh use

sudo systemctl restart jenkins

**🔹 Click: Manage Roles**

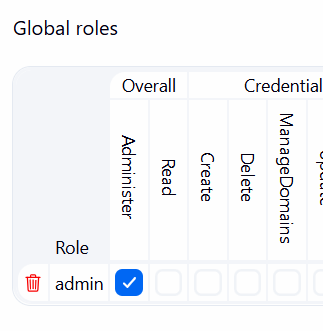
**A) Global Roles (sabse important)**

**🛡 Admin Role**

* Role Name: admin

Reason: Administer permission dene se role effectively admin ban jata ha

* Permissions (tick karo):
  + Overall section m → Administer ko tick✔ karoh which is already done see the ss



Then baki k bachay hua saray box autoselected ho jayegay but hua nahi bcz

**❌ Jenkins ab baaki permissions auto-select nahi karta**

Purane tutorials / blogs me likha hota hai:

“Administer tick karte hi sab auto-select ho jata hai”

👉 **Naye Jenkins versions me ye behavior change ho chuka hai**.

Isliye:

* Sirf **Overall → Administer ✔** dikh raha hai
* Baaki boxes blank reh rahe hain
* **YEH 100% OK HAI** ✅

now:

**Dashboard → Manage Jenkins → Security → Manage and Assign Roles k baad go**

Global roles section me aao then role to add wale box m likho: developer then add button p click karoh. Then

Developer role ke permission set karoh;

Aisay karna hai



Is box ko quick verification say fill karenagy.

**✅ QUICK VERIFICATION; har box m tick karoh jaisy batara hai jinpay nhi bola unko khali chord doh**

**🔹 Overall**

* ✔ **Read**
* ❌ Administer  
  👉 **Correct**

**🔹 Credentials**

* ✔ Create, ✔ Update
* ❌ Delete, ❌ ManageDomains nahi karna baki sab p kardoh   
  👉 **Correct (secure)**

**🔹 Agent**

* ✔ Build, ✔ Connect
* ❌ Provision, ❌ Delete  
  👉 **Correct**

**🔹 Job**

* ✔ Create, ✔ Configure, ✔ Build, ✔ Read, ✔ Workspace
* ❌ Delete, ❌ Move  
  👉 **Correct**

**🔹 Run:** ismay replay per tick nahi bhi hota toh kio dikkat nahi hai but ismay delete and update per tick nahi karna that means pura khaali chordoh isay

**🔹 View**

* ✔ Read, ✔ Create
* ❌ Delete, ❌ Configure  
  👉 **Correct**

**🔹 SCM**

* ✔ Tag  
  👉 **Correct**

**🔹 Metrics**

* ✔ HealthCheck  
  👉 **Correct but wait.**

**Save per click karnay say phele ctrl+5 say refresh karkay then go**

**Manage Jenkins → Security → Security then scroll down**

**1️⃣ Default Crumb Issuer select rehne do  
2️⃣ Enable proxy compatibility ✔ tick karo  
3️⃣ Page ke niche Save click karo**

Agar hum default crumb issue enable proxy compatibility per tick nahi kartay toh hmay 403 naam ka error milta developer permission set karkay bos fill karnay k baaad jab save kartay toh 403 error milta hai.ab toh humnay karliya but kabhi jab future new Jenkins start kaarenagy tohi say karna hoga ,

**Then Manage Jenkins → Security → Manage and Assign Roles**

**Global role m >> add may developer likar save kardoh > then vaapis developer permission set karoh that means vox vaapis fill karoh upper jakay quick verification say then save karoh ho jayega . ab 403 error nahi ayega .easliy save ho jayega .**

**Manage role done now we do**

**>>>>> nichay jayoh now we do assign role**

**Assign Roles (Users ko roles dena):**

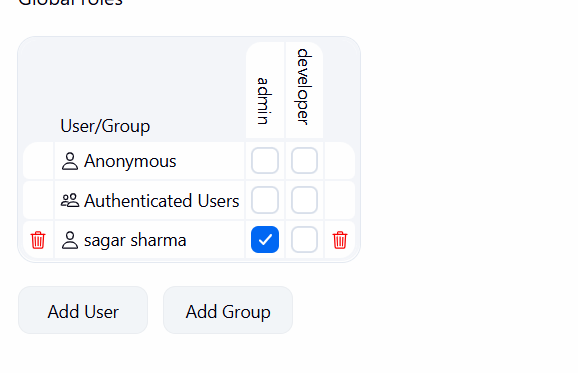
Vahi same

Manage Jenkins → Security → Manage and Assign Roles

**🔹 Click: Assign Roles**

**A) Global roles section**

**Step 1: Global roles section:**

****Tumhare screenshot me:

* User: **sagar sharma**
* Roles: **admin | developer**

**✅ Kya karna hai**

**Agar tum khud admin ho (recommended):**

* ☑ **admin** ✔
* ⬜ developer (optional)

👉 **Current SS me ye already correct hai** 👍

**Agar developer assign karna ho:**

* Us user ke saamne
* ☑ **developer** tick karo
* ⬜ admin unchecked

Example:

sagar sharma → developer ✔

**🔹 Step 2: Anonymous & Authenticated Users**

**❌ Kya NAHI karna**

* ❌ Anonymous ko koi role mat do
* ❌ Authenticated Users ko admin mat do

(Ye security risk hai)

👉 Tumhare SS me dono blank hain — **perfect** ✅

**🔹 Step 3: Save (MOST IMPORTANT)**

Page ke niche:

Save

📸 Screenshot me ye dikhna chahiye:

* Username: **sagar sharma**
* Role tick: **admin** ya **developer**
* Anonymous blank

3.now we learn Credentials Management :

Credentials kya hote hain?

**ANS:** Credentials wo secret details hoti hain jinke through  
👉 kisi system, server, app, ya service me login ya access milta hai.

Matlab:

“Tum kaun ho” prove karne ka tareeqa = Credentials

Common types of credential:

Password , ssh keys , token etc

>>Role of credential in Jenkins:

Jenkins me Credentials ka role hai secure authentication provide karna, taaki Jenkins bina password expose kiye external systems ke saath safely connect kar sake.

Jenkins credentials github repo acess , srever(ec2) login , docker , sonakrqube ,aws ascess use karta hai

➕ Credentials add kaise karein:

Manage Jenkins → security section k andar >>

Credentials → System → Global → Add Credentials

Step by step:

**1️⃣ Jenkins Dashboard open karo**

Browser me Jenkins URL open karo

**2️⃣ Manage Jenkins par jao**

Left side menu → **Manage Jenkins**

**3️⃣ Credentials section open karo**

Manage Jenkins → **Credentials**

**4️⃣ System → Global select karo**

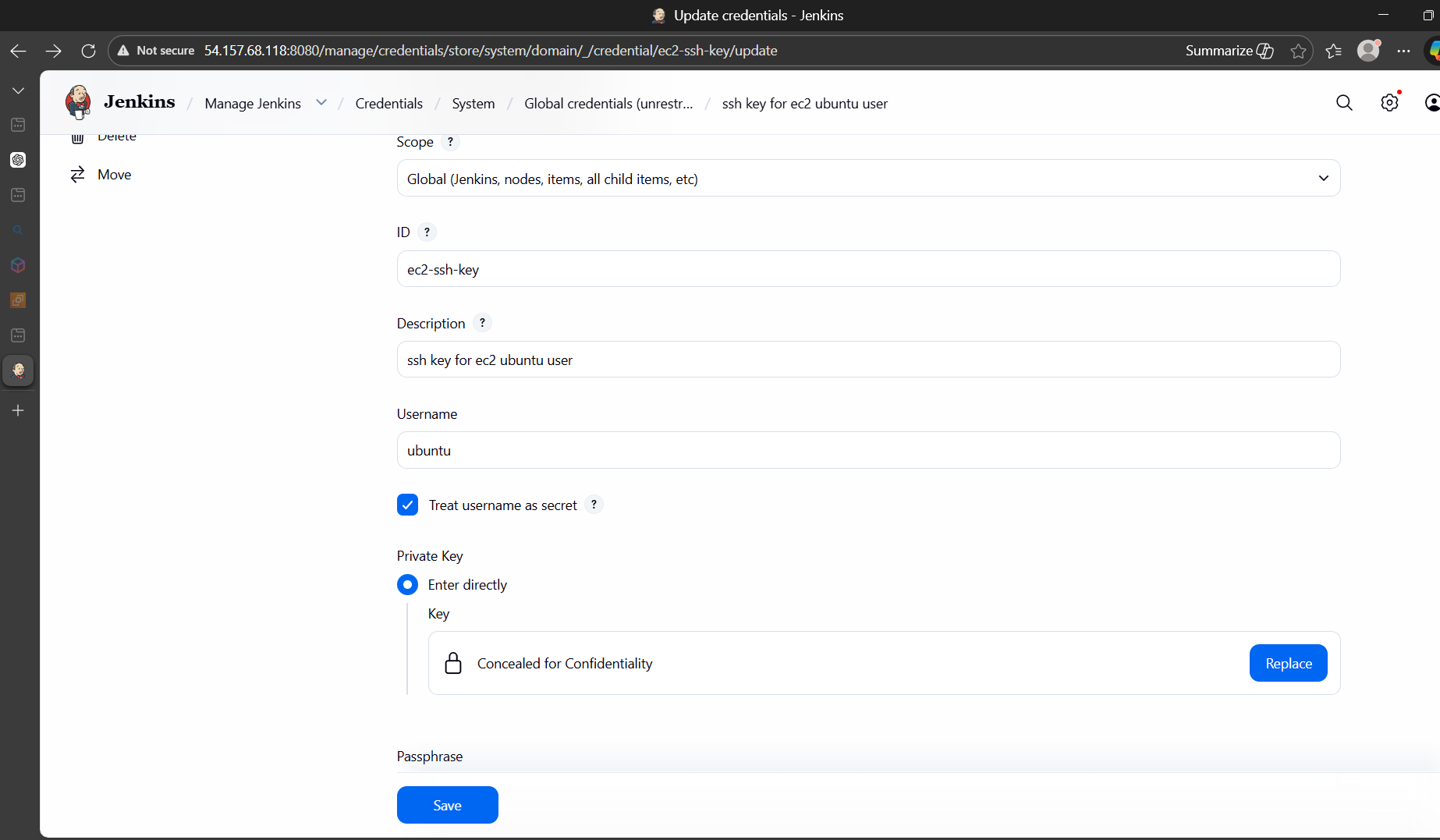
* **System** = Jenkins internal scope
* **Global** = sab jobs ke liye available

5️⃣ Add Credentials par click karo

**6️⃣ Credentials details fill karo**

**🧪 Example: GitHub SSH Key**

* **Kind**: SSH Username with private key

****

Private key kaisy nikalenagy >>file m search karoh ec2.pem which is our currentkeypair of current instance ec2.pem search karkay downloads m usko pura copy karkay yha paste kardoh.

Passphrase ko empty chord saktayhai bcz ec2.pem m kio passphrase nahi hota .

**7️⃣ Save button dabao**

* Credentials successfully add ho jayenge ✅

**>>> ab Jenkins agent sikhengay>>>>**

**(4) Jenkins Agents (Slave / Worker Nodes):**

🧠 Agent kya hai?

Jenkins Agent ek worker machine hoti hai jahan actual build / test / deploy tasks run hote hain.  
Jenkins ka Master (Controller) sirf:

* jobs schedule karta hai
* configuration manage karta hai
* results collect karta hai

👉 Heavy kaam agents pe chalta hai, master pe nahi.

>>Jenkins master ky hota hai:

Ans: Master = brain (control &manage)

Agent = hands (actual kaam)

Master say Jenkins fast , scalable and secure banta hai.

>>Jenkins agent kyu use kartay hau?

Ans: i) Multiple builds **parallel** chalane ke liye

ii) Different OS par builds (Linux / Windows)

iii) Master ka **load kam** rakhne ke liye

iv) Production-like environment me build/test

**>> Jenkins agent kaisy use kartay hai?**

**Ans: step-1: agent machine ready karoh:**

Type these both commands in vs code terminal : ( i) sudo apt update

Agar is command k baad error aye toh yeh karoh

**>>Step1: jenkins repo ko disable karoh (comment out):**

sudo mv /etc/apt/sources.list.d/jenkins.list /etc/apt/sources.list.d/jenkins.list.disabled

**expected output: no output**

matlab Jenkins repo temporariliy off hogya

**step2: apt clean +update;**

sudo apt clean

sudo apt update

>>>ab error nahi ayega like

NO\_PUBKEY 7198F7D14ABFC68

ISKA MATLAB: Jenkins already install + running hai → repo ki zarurat **abhi nahi hai**

(ii) sudo apt install openjdk-17-jdk -y

**Step 2: Jenkins me SSH credential add karo**:

Manage Jenkins → Credentials → System → Global → Add Credentials

Kind: **SSH Username with private key**

Username: ubuntu

Private key: .pem ka content already done

**Step 3: New Agent create karo:**

Manage Jenkins → Nodes → New Node

Fill details:

* Name: linux-agent
* Type: **Permanent Agent**
* Remote root dir: /home/ubuntu/jenkins
* Labels: linux
* Usage : using this node as much as possible (isi ko use karoh mostly)

**Usage k andar 2 option hotay hai:**

**(1)Use this node as much as possible** ✅: iska matlab Jenkins is node ko default agent ki tarah use Karega.

Means Agar job me kio label mention nahi hai toh Jenkins isi node per job chla dega

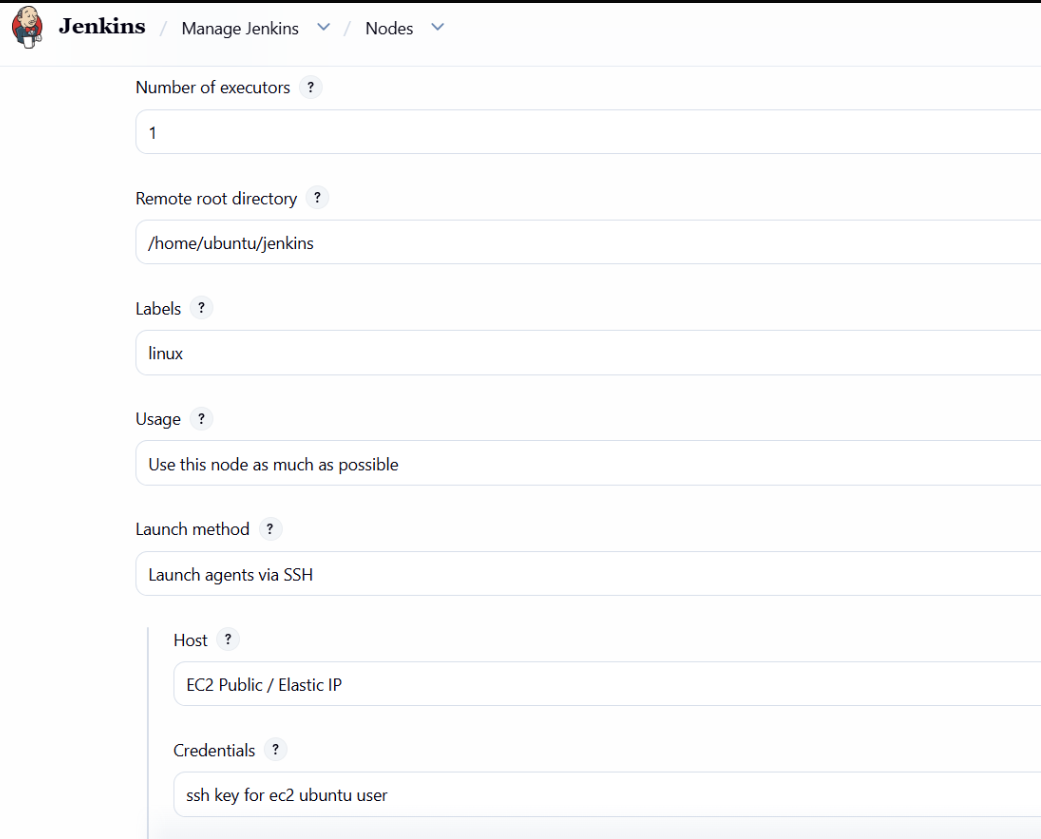
(2)**Only build jobs with label expressions matching this node** ❌:

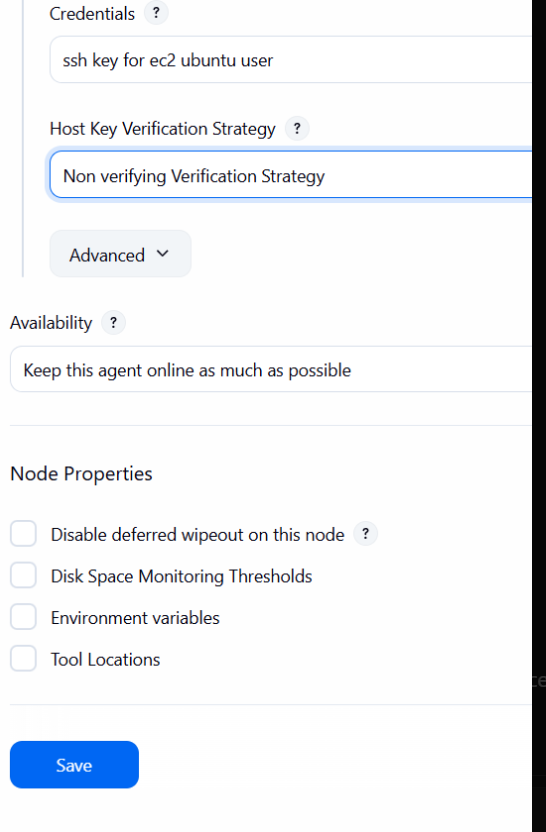
iska matlab Jenkins sirf wahi jobs is node par chalayega:

* Jinke job config / pipeline me label diya hoga >> ab nichay jayoh >>

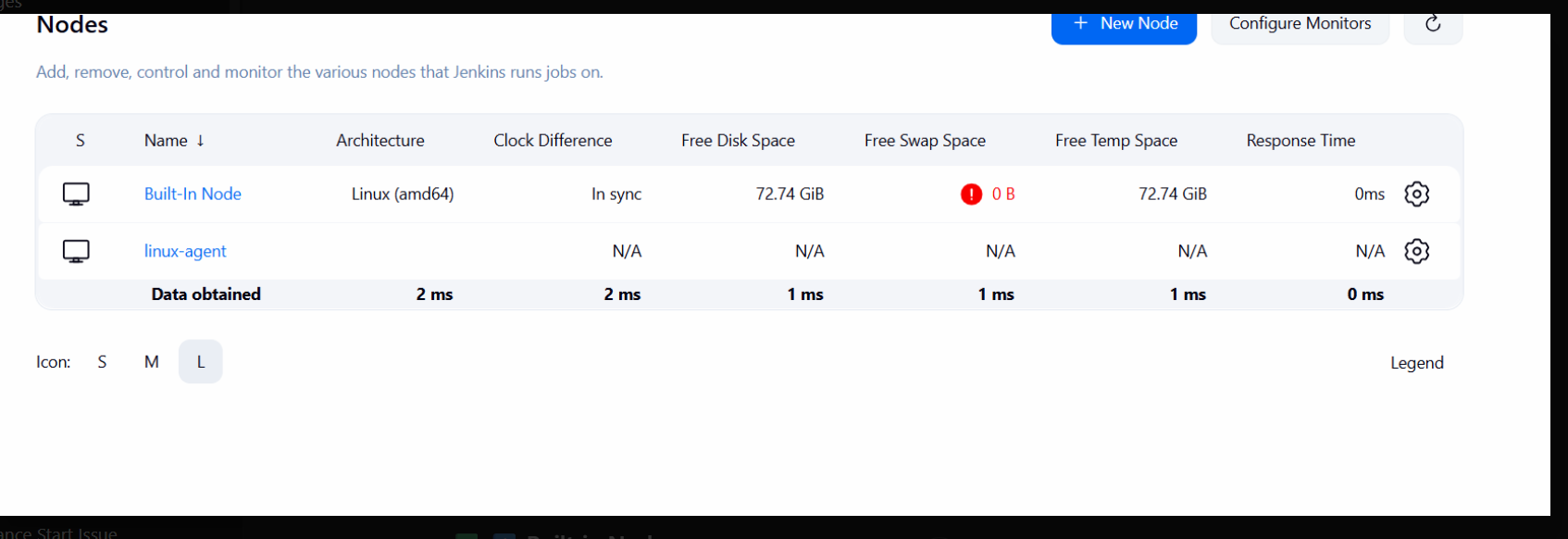
**Step 4: Launch method:**

* Launch agent via SSH
* Host: EC2 Public / Elastic IP
* Credentials: ssh key for ec2 ubuntu user
* Host Key Verification: Non verifying





👉 Save → Agent connected ✅ iskay baad page khula toh



But error aya

hmara bulit node toh thik hai yeh red line ignore kar saktay hai . kio block nahi hai

but linux-agent :

architecture : N/A

Disk /temp / response time: N/A

>>Iska matlab:Agent Jenkins se SSH ke through connect nahi ho pa raha  
ya phir abhi tak successfully launch hi nahi hua.

Agar sab sahi hota toh ;

linux-agent ke saamne aisa dikhta:

* Architecture: Linux (amd64)
* Free Disk Space: kuch GB
* Response time: kuch ms  
  👉 Tab kehte **Agent ONLINE hai** ✅

**>>Ab linux agent ko connect kartay hai aur fix kartay hai:**

**Run some commandsvs code in;**

sudo systemctl status ssh >>>expexted output: running

java -version >>> like 45.78

ls -ld /home/ubuntu/Jenkins

**agar iska ouput aye ls:cannot acess '/home/ubuntu/jenkins': No such file or directory**

Matlab :Jenkins agent ka Remote root directory exist hi nahi karta, isliye agent connect hote hi crash ho jaata hai. then type

sudo mkdir -p /home/ubuntu/jenkins

sudo chown -R ubuntu:ubuntu /home/ubuntu/jenkins

sudo chmod -R 755 /home/ubuntu/Jenkins

verify : ls -ld /home/ubuntu/Jenkins

**expected ouput: drwxr-xr-x 2 ubuntu ubuntu 4096 Feb 7 /home/ubuntu/jenkins**

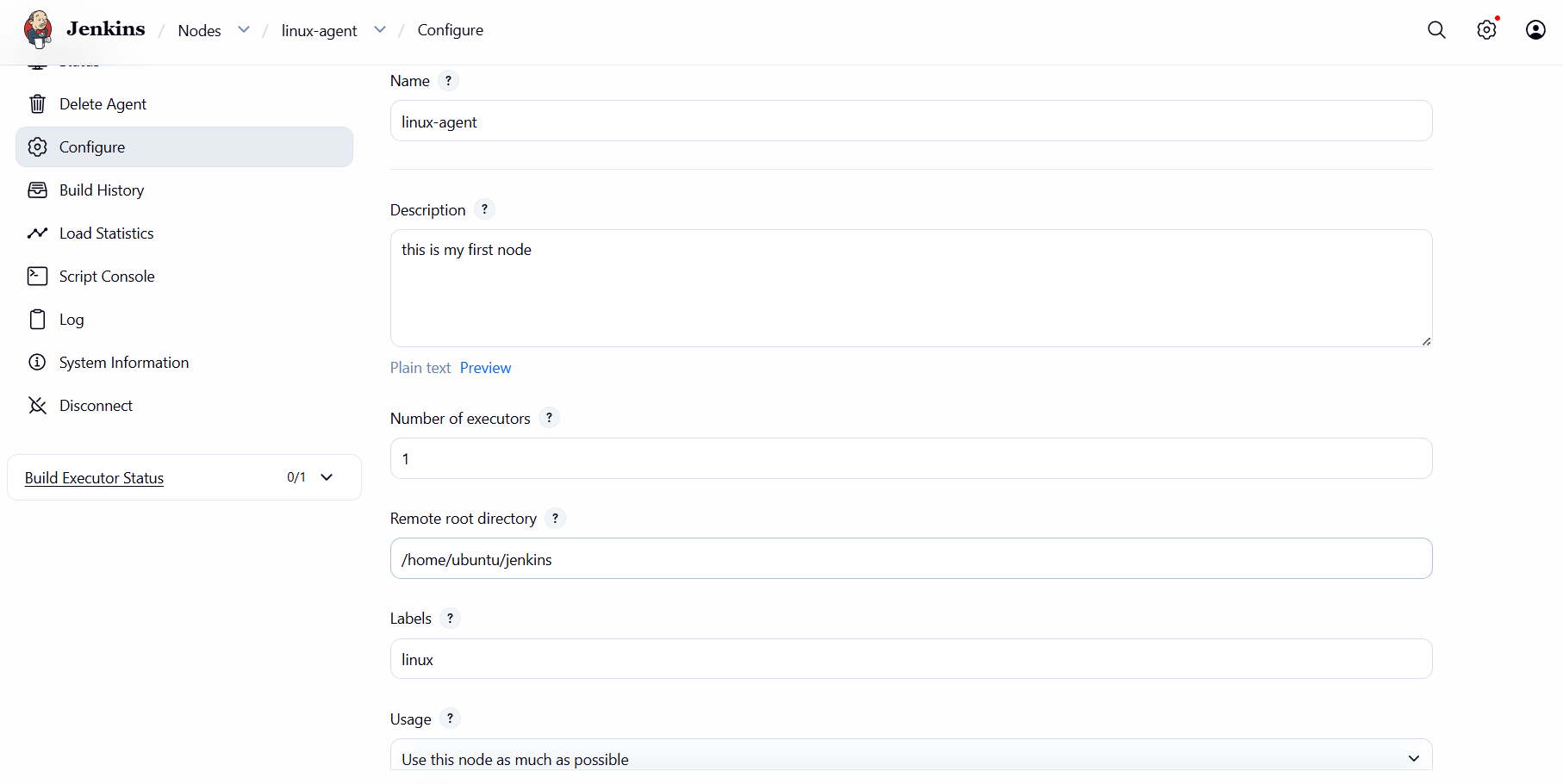
ab manage Jenkins >> nodes >>

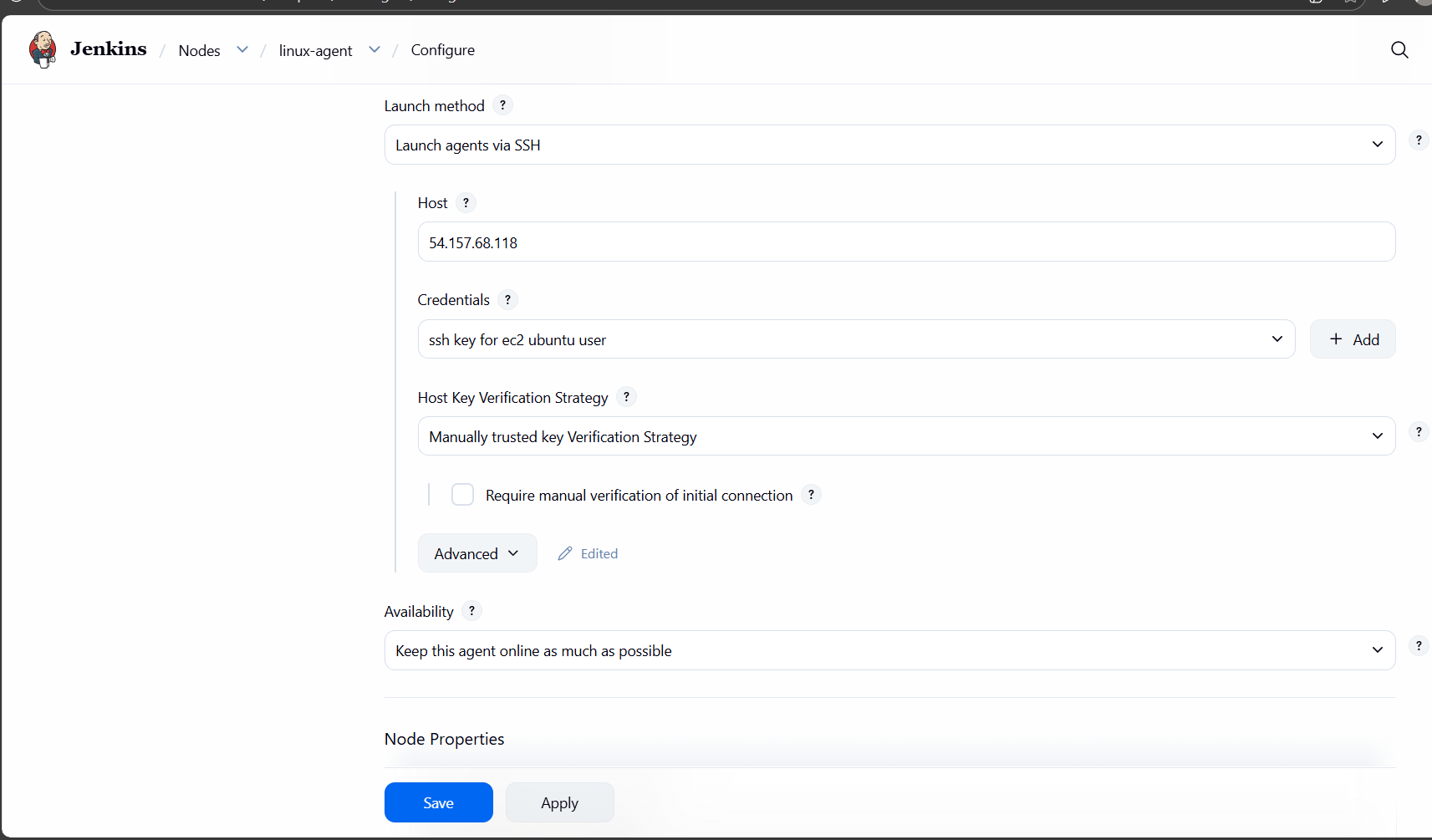
linux agent per click karoh

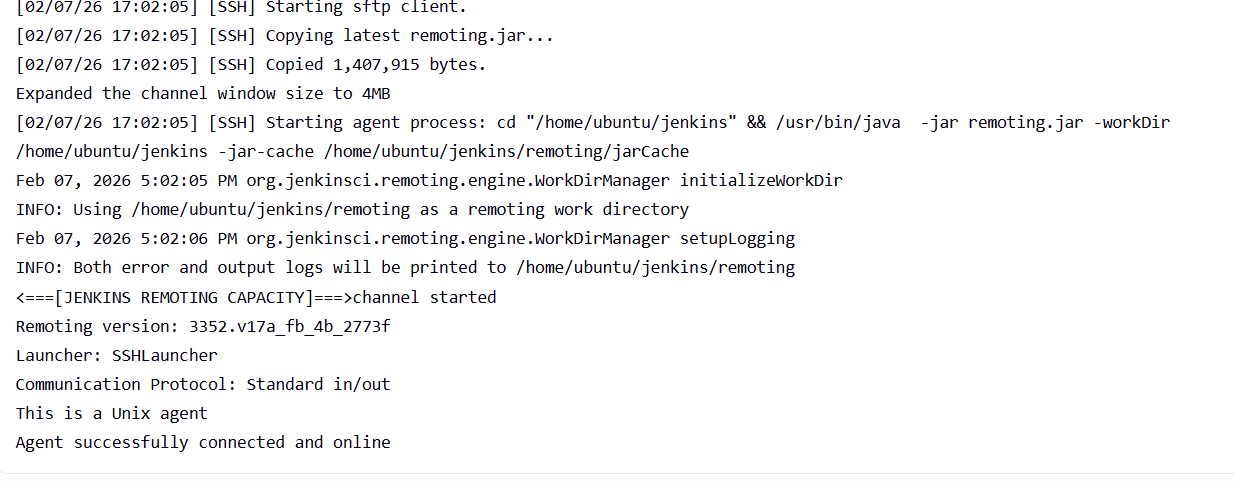
**Node config (ALL CORRECT)**

* Number of executors: 1
* Remote root directory: /home/ubuntu/jenkins ✅
* Labels: linux
* Usage: **Use this node as much as possible** ✅
* Launch method: **Launch agents via SSH**
* Host: **54.157.68.118(currenet instance elastic ip)**
* Credentials: **ssh key for ec2 ubuntu user**
* Port: 22
* Host Key Verification:
  + *Manually trusted* (OK)
* Availability:
  + Keep this agent online as much as possible

👉 **Ye sab bilkul theek hai then save then click log you will see ssh connection successful**

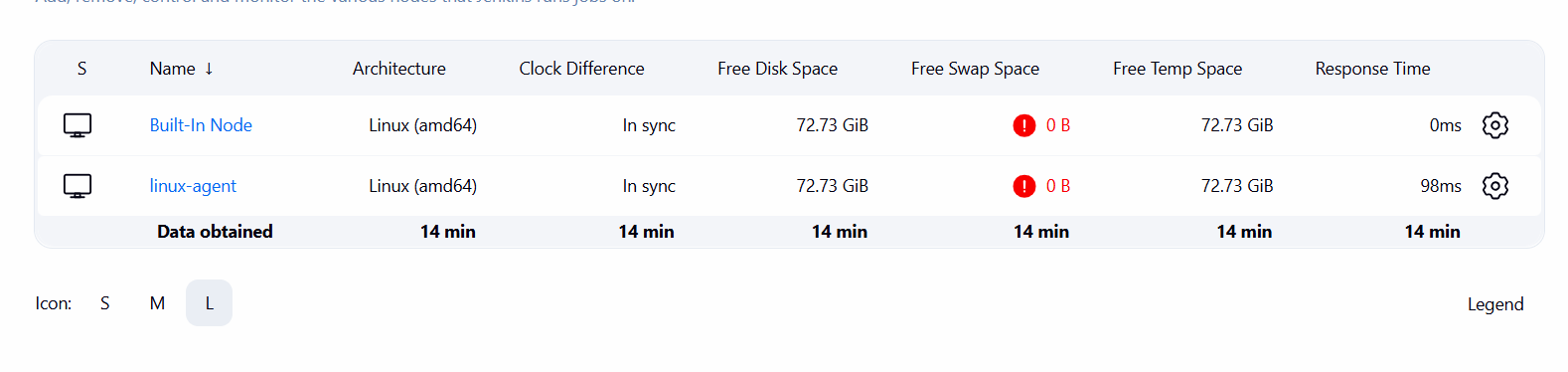
****

****

****

Browser me Ctrl + F5

* Old log clear ho jayega
* Sirf latest status dikhega

****

Jo yeh free swap per red hai isko ignore kar saktay hai kio dikkat nhi hai

**🧪 Agent test kaise karein?**

Item banyoh:Jenkins-test-node-pipeline-demo then click on pipeline then ok the click on pipeline then pipeline script m code paste kardoh

aur haa agar url niklanay k liye github open karoh then profile per click karoh the n repository then sagar-git-demo-2 per click then code per click per click karogay to url mil jayega >ab yeh code copy thenpasteinscrip-box:

pipeline {

agent { label 'linux' }

stages {

stage('Checkout') {

steps {

git branch: 'main',

url: 'https://github.com/sagarvashist19-boop/jenkins-git-demo-2.git'

}

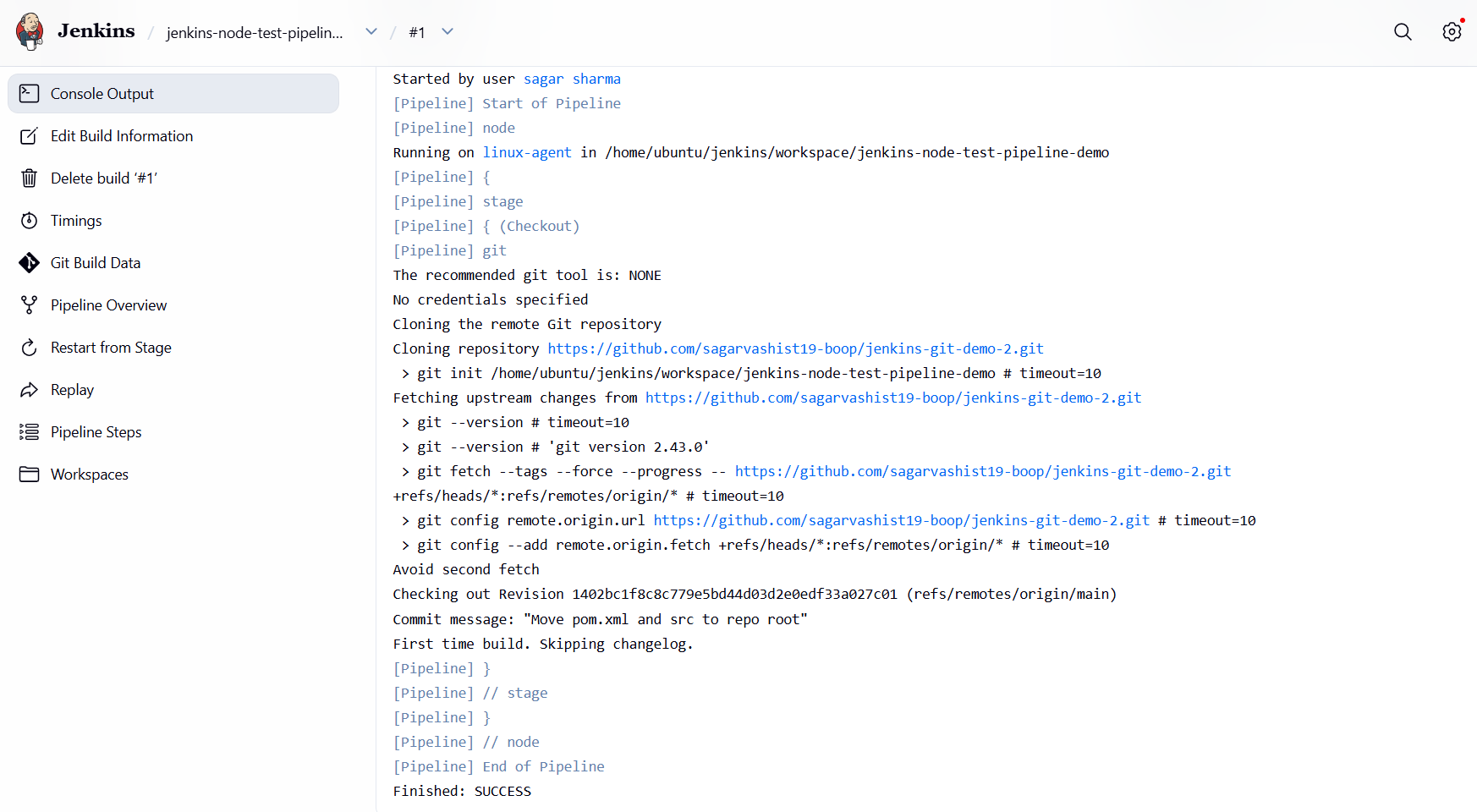
}

}

}

**Is pipeline me kya ho raha hai (line-by-line, simple):**

* agent { label 'linux' }  
  👉 Job **linux-agent** par chalegi (jo tumne abhi successfully connect kiya 👍)
* stage('Checkout')  
  👉 GitHub se code pull karega
* git branch: 'main'  
  👉 main branch checkout hogi



🎯 FINAL GOAL ACHIEVED

✅ Jenkins secure  
✅ Credentials protected  
✅ Agents connected  
✅ Master load kam  
✅ Real-world DevOps setup

**🔴 Day 10 – Jenkins Real Projects + Interview**

**Final day of Jenkins**

**>>Jenkin+docker ky hota hai?**

**Ans:** Jenkins Docker ka use karke application ko container me build & run karta hai.

👉 Matlab:

* “Mere system pe chal raha hai” wali problem khatam
* Same app har server pe same chalegi

**>>Kyun use karte hain?**

* Environment issue nahi hota
* Fast deployment
* Production jaisa environment locally / CI me

**🔹 Kahan use hota hai?**

* Java / Node / Python apps
* Microservices
* Real companies me **almost mandatory**
* **🔹 Example (real flow)**
* **GitHub → Jenkins → Docker build → Docker run**

**🔹 Jenkinsfile snippet**

**pipeline {**

**agent any**

**stages {**

**stage('Build Docker Image') {**

**steps {**

**sh 'docker build -t myapp .'**

**}**

**}**

**stage('Run Container') {**

**steps {**

**sh 'docker run -d -p 8081:8080 myapp'**

**}**

**}**

**}**

**}**

**Iska matlab:**

**🔵 pipeline {**

👉 Jenkins Declarative Pipeline ka start.  
Iske andar poora CI/CD flow likha hota hai**.**

**agent any**

👉 Jenkins ko bol rahe ho:

“Job kisi bhi available node / machine pe chala do” condition: us node p docker installed hona chaiye

**stages {**

👉 Pipeline ko steps / phases me divide karta hai**.**

**stage('Build Docker Image')**

👉 Docker image banane ka stage**.**

**steps {**

👉 Is stage me kya-kya commands chalengi**.**

**sh 'docker build -t myapp .'**

👉 Shell command:

* docker build → image banana
* -t myapp → image ka naam = myapp
* . → current directory me jo Dockerfile hai, use kare

📌 Matlab:

Jenkins workspace me Dockerfile se Docker image ban rahi hai

**stage('Run Container')**

👉 Banayi hui image se container start karne ka stage

**sh 'docker run -d -p 8081:8080 myapp'**

👉 Docker command:

* docker run → container start
* -d → background me run (detached)
* -p 8081:8080
  + Host port 8081
  + Container port 8080
* **myapp → image ka naam**

📌 Matlab:

Browser se http://<server-ip>:8081 pe app access ho jaayegi

**🧠 Complete Flow (samajhne ke liye)**

**Code + Dockerfile**

**↓**

**Docker Image (myapp)**

**↓**

**Docker Container run**

**↓**

**App live on port 8081**

**📌 Interview line:**

* Jenkins Docker ko use karke application ko containerized environment me build aur deploy karta hai.

**🔹 Jenkins + AWS kya hota hai?**

Jenkins AWS EC2 server par automatically code deploy karta hai**.**

**🔹 Kyun use hota hai?**

* Manual login band
* Auto deployment
* Production ready setup

**🔹 Kahan use hota hai?**

* Web apps
* Backend APIs
* Live production servers
* **🔹 Real flow;**
* GitHub → Jenkins → Build → SSH → EC2 → Deploy

**🔹 Jenkinsfile example;**

stage('Deploy to EC2') {

steps {

sshagent(['ec2-ssh-key']) {

sh '''

ssh ubuntu@EC2\_IP "

cd /home/ubuntu/app &&

git pull &&

docker restart app

"

'''

}

}

}

Code means:

**stage('Deploy to EC2')**

👉 Jenkins pipeline ka **deployment stage**  
Iska kaam: **EC2 server par application update karna**

**steps {**

👉 Is stage me jo commands chalengi, wo yahan likhi hoti hain.

**sshagent(['ec2-ssh-key']) {**

👉 Jenkins apna **SSH credential load** karta hai.

* ec2-ssh-key = Jenkins me save ki hui SSH key
* Is block ke andar:
  + SSH login **password ke bina**
  + Secure tareeke se hota hai
* **sh '''**
* 👉 Multi-line **shell script** start  
  (Taaki ek se zyada commands likh sako)

**ssh ubuntu@EC2\_IP "**

👉 Jenkins EC2 machine me **SSH login** karta hai

* ubuntu → EC2 ka username
* EC2\_IP → EC2 ka public IP / DNS

📌 Matlab:

Jenkins remote server pe commands bhej raha hai

**cd /home/ubuntu/app &&**

👉 EC2 server par:

* Application ke folder me jaana

&& ka matlab:

Agar ye command successful ho, tabhi next chale

**git pull &&**

👉 Latest code GitHub se pull karta hai

📌 Matlab:

Server par **naya code update**

**docker restart app**

👉 Docker container ko restart karta hai

* app = container ka naam

📌 Matlab:

New code ke saath application dobara start

**"**

👉 SSH command ka end

**'''**

👉 Multi-line shell script end

**🧠 Complete Flow (samajhne ke liye)**

Jenkins

↓ (SSH)

EC2 Server

↓

Go to app folder

↓

Pull latest code

↓

Restart Docker container

**3️⃣ Jenkins Best Practices:**

**✅ Best practices kya hoti hain?**

Industry ke **rules** jo Jenkins ko secure & scalable banate hain.

**🔹 Important best practices**

* Jenkinsfile repo me rakho
* Credentials hardcode mat karo
* Master pe build mat chalao
* Agents use karo
* Small stages banao
* 📌 **Interview question**
* Jenkins best practices kya hain?
* ✔️ Answer:
* Jenkinsfile, credentials management, master-agent architecture, small pipelines.
* **4️⃣ End-to-End CI/CD Project :**
* **🔹 CI/CD kya hota hai?**
* **CI** → Code build + test
* **CD** → Auto deploy

**🔹 Full real project flow**

Developer

↓

GitHub (code push)

↓

Webhook

↓

Jenkins Pipeline

↓

Build (Maven)

↓

Docker Image

↓

Deploy on EC2

<<<<<<<<<<iska code hai>>>

pipeline {

agent { label 'linux' }

tools {

maven 'Maven-3.8.7'

}

stages {

stage('Checkout') {

steps {

git branch: 'main',

url: 'https://github.com/sagarvashist19-boop/jenkins-git-demo-2.git'

}

}

stage('Build (Fast)') {

steps {

sh 'mvn clean package -DskipTests'

}

}

stage('Docker Build') {

steps {

sh 'docker build -t jenkins-demo-app:latest .'

}

}

stage('Docker Deploy') {

steps {

sh '''

docker rm -f jenkins-demo-app || true

docker run -d --name jenkins-demo-app -p 8081:8080 jenkins-demo-app:latest

'''

}

}

}

post {

success {

echo '✅ CI/CD PIPELINE SUCCESSFUL'

}

failure {

echo '❌ PIPELINE FAILED – CHECK LOGS'

}

}

}

**🧠 Ye pipeline EXACTLY kya karegi? (Short & Clear)**

1️⃣ **Checkout**  
→ Tumhare GitHub repo se latest code lega

2️⃣ **Build (Fast)**  
→ mvn clean package -DskipTests  
→ build fast (tests skip)

3️⃣ **Docker Build**  
→ Docker image banegi:  
jenkins-demo-app:latest

4️⃣ **Docker Deploy**  
→ Purana container remove  
→ Naya container run  
→ App live on:

http://<server-ip>:8081

**⚠️ IMPORTANT CHECKLIST (Run karne se pehle)**

✔ Repo me **Dockerfile** hona chahiye  
✔ Jenkins user ke paas **Docker permission** ho  
✔ Port **8081 free** ho  
✔ Maven tool Jenkins me configured ho

**🎯 Interview / Viva One-Liner**

👉 Ye Jenkinsfile GitHub se code fetch karta hai, Maven se fast build karta hai, Docker image banata hai aur container run karke application deploy karta hai.

**5️⃣ Jenkins vs GitHub Actions**

| **Jenkins** | **GitHub Actions** |
| --- | --- |
| Self-hosted | GitHub managed |
| Powerful & flexible | Easy & limited |
| Plugins support | YAML based |
| Industry standard | GitHub focused |

📌 Interview answer:

Jenkins zyada powerful aur enterprise level tool hai.

**6️⃣ CI vs CD**

| **CI** | **CD** |
| --- | --- |
| Build + test | Deploy |
| Code quality | Delivery |
| Fast feedback | Auto release |

📌 Interview one-liner

CI code ko test karta hai, CD usko deploy karta hai.

**7️⃣ Jenkinsfile**

**🔹 Jenkinsfile kya hota hai?**

Pipeline ka **code version** (IaC concept)

**🔹 Kyun use hota hai?**

* Automation
* Version control
* Repeatable builds

📌 Interview:

Jenkinsfile pipeline ko code ke form me define karta hai.

**8️⃣ Pipeline Stages**

**🔹 Stage kya hota hai?**

Pipeline ka ek step

**🔹 Example**

* Checkout
* Build
* Test
* Deploy

📌 Interview:

Stages pipeline ko readable aur manageable banate hain.

**9️⃣ Jenkins Master–Agent:**

**🔹 Master**

* UI
* Job scheduling

**🔹 Agent**

* Actual build
* Heavy work

📌 Interview:

Master manage karta hai, agent execute karta hai.

**🔟 Webhook Working (VERY IMPORTANT 🔥)**

**🔹 Webhook kya hota hai?**

GitHub ka Jenkins ko **auto signal** dena

**🔹 Flow**

git push → GitHub → webhook → Jenkins → pipeline

📌 Interview:

Webhook Jenkins ko real-time trigger karta hai.

**🎯 Interview Ready One-Line Summary:**

Jenkins ek CI/CD automation tool hai jo GitHub, Docker aur AWS ke sath integrate hoke end-to-end deployment pipelines, agents aur webhooks. Now we learn interview question

**✅ PART 1: Jenkins Interview Q&A (30+ Questions)**

**🔹 Jenkins Basics**

**1️⃣ Jenkins kya hai?**

**Answer:**  
Jenkins ek open-source CI/CD automation tool hai jo build, test aur deploy process ko automate karta hai.

**2️⃣ CI/CD kya hota hai?**

**Answer:**  
CI code ko automatically build & test karta hai, aur CD code ko automatically deploy karta hai.

**3️⃣ Jenkins ka use kyun hota hai?**

**Answer:**  
Manual work kam karne, fast delivery aur error-free deployment ke liye.

**4️⃣ Jenkins ka alternative kya hai?**

**Answer:**  
GitHub Actions, GitLab CI, Azure DevOps, CircleCI.

**🔹 Jenkins Architecture**

**(1) Jenkins Master aur Agent kya hota hai?**

**Answer:**  
Master jobs manage karta hai, agent actual build execute karta hai.

**(2) Jenkins Agent kyun use karte hain?**

**Answer:**  
Load distribute karne, scalability aur different environments me build ke liye.

**(3) Jenkins built-in node kya hota hai?**

**Answer:**  
Wahi machine jahan Jenkins install hota hai.

**🔹 Jenkins Pipeline**

**(1) Jenkins Pipeline kya hoti hai?**

**Answer:**  
Pipeline Jenkins jobs ko code ke form me define karti hai (Jenkinsfile).

**(2) Jenkinsfile kya hota hai?**

**Answer:**  
Ek text file jo CI/CD process ko define karti hai.

**(3) Declarative vs Scripted Pipeline?Answer:**  
Declarative simple hoti hai, Scripted flexible hoti hai.

**(4) Pipeline stages kya hoti hain?**

**Answer:**  
Pipeline ke logical steps jaise Checkout, Build, Test, Deploy.

**(5) agent none kya karta hai?**

**Answer:**  
Jenkins master pe executor waste nahi karta, har stage apna agent choose karta hai.

**🔹 Git & Webhooks**

**1️⃣Jenkins GitHub se kaise trigger hota hai?**

**Answer:**  
Webhook ke through.

**(2)Webhook kya hota hai?**

**Answer:**  
GitHub ka Jenkins ko real-time notification dena.

**(3) Poll SCM vs Webhook?**

**Answer:**  
Polling time-based hota hai, webhook event-based hota hai (better).

**🔹 Credentials & Security**

**1️⃣ Jenkins credentials kya hote hain?**

**Answer:**  
Passwords, SSH keys, tokens jo secure access ke liye use hote hain.

**(2) Jenkins credentials secure kaise hote hain?**

**Answer:**  
Encrypted form me store hote hain.

**(3) SSH credentials ka use kahan hota hai?**

**Answer:**  
Agent connect, GitHub SSH aur server deploy ke liye.

**🔹 Docker & Deployment**

**1️⃣ Jenkins + Docker kyun use hota hai?**

**Answer:**  
Application ko containerize karke consistent deployment ke liye.

**2️⃣ Docker image aur container me difference?**

**Answer:**  
Image template hoti hai, container running instance hota hai.

**(3) Jenkins EC2 pe deploy kaise karta hai?Answer:**  
SSH ke through ya Docker containers ke through.

**🔹 Best Practices**

**(1) Jenkins best practices kya hain?**

**Answer:**  
Jenkinsfile use karna, credentials hardcode na karna, agents use karna

(2) **Jenkinsfile repo me kyun rakhte hain?**

**Answer:**  
Version control aur repeatable builds ke liye.

**(3) Jenkins me labels ka use?**

**Answer:**  
Specific agent par job run karne ke liye.

**🔹 Troubleshooting:**

**(1)Jenkins agent offline kyun hota hai?**

**Answer:**  
SSH issue, Java missing, directory missing, wrong credentials.

**2️⃣ Jenkins port 8080 use kyun karta hai?**

**Answer:**  
Web UI access ke liye default port.

**(3) Jenkins restart command?**

**Answer:**  
sudo systemctl restart Jenkins

**🔹 Comparison**

**1) Jenkins vs GitHub Actions?**

**Answer:**  
Jenkins zyada flexible & enterprise-level tool hai.

**(2) Jenkins vs Ansible?**

**Answer:**  
Jenkins CI/CD ke liye, Ansible configuration management ke liye.

**(3) Jenkins future?**

**Answer:**  
Jenkins still widely used with Docker & Kubernetes.

📌 Matlab:

Jenkins ab EC2 me SSH kar sakta hai